

S-PLUS[®] Workbench User's Guide

May 2007

Insightful Corporation
Seattle, Washington

**Proprietary
Notice**

Insightful Corporation owns both this software program and its documentation. Both the program and documentation are copyrighted with all rights reserved by Insightful Corporation.

The correct bibliographical reference for this document is as follows:

S-PLUS[®] Workbench User's Guide, Insightful Corporation, Seattle, WA.

Printed in the United States.

Copyright Notice

Copyright © 1987-2007, Insightful Corporation. All rights reserved.

Insightful Corporation
1700 Westlake Avenue N, Suite 500
Seattle, WA 98109-3044
USA

Trademarks

Insightful, Insightful Corporation, the Insightful logo and tagline "the Knowledge to Act," Insightful Miner, S+, S-PLUS, S+FinMetrics, S+EnvironmentalStats, S+SeqTrial, S+SpatialStats, S+Wavelets, S+ArrayAnalyzer, S-PLUS Graphlets, Graphlet, Trellis, and Trellis Graphics are either trademarks or registered trademarks of Insightful Corporation in the United States and/or other countries. Intel and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries. Microsoft, Windows, MS-DOS and Windows NT are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Sun, Java and Solaris are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States or other countries. UNIX is a registered trademark of The Open Group. Linux is a registered trademark of Linus Torvalds in the United States and other countries.

ACKNOWLEDGMENTS

S-PLUS would not exist without the pioneering research of the Bell Labs S team at AT&T (now Lucent Technologies): John Chambers, Richard A. Becker (now at AT&T Laboratories), Allan R. Wilks (now at AT&T Laboratories), Duncan Temple Lang, and their colleagues in the statistics research departments at Lucent: William S. Cleveland, Trevor Hastie (now at Stanford University), Linda Clark, Anne Freeny, Eric Grosse, David James, José Pinheiro, Daryl Pregibon, and Ming Shyu.

Insightful Corporation thanks the following individuals for their contributions to this and earlier releases of S-PLUS: Douglas M. Bates, Leo Breiman, Dan Carr, Steve Dubnoff, Don Edwards, Jerome Friedman, Kevin Goodman, Perry Haaland, David Hardesty, Frank Harrell, Richard Heiberger, Mia Hubert, Richard Jones, Jennifer Lasecki, W.Q. Meeker, Adrian Raftery, Brian Ripley, Peter Rousseeuw, J.D. Spurrier, Anja Struyf, Terry Therneau, Rob Tibshirani, Katrien Van Driessen, William Venables, and Judy Zeh.

S-PLUS BOOKS

The S-PLUS[®] documentation includes books to address your focus and knowledge level. Review the following table to help you choose the S-PLUS book that meets your needs. These books are available in PDF format in the following locations:

- In your S-PLUS installation directory (**SHOME\help** on Windows, **SHOME/doc** on UNIX/Linux).
- In the S-PLUS Workbench, from the **Help ► S-PLUS Manuals** menu item.
- In Microsoft[®] Windows[®], in the S-PLUS GUI, from the **Help ► Online Manuals** menu item.

S-PLUS documentation.

Information you need if you...	See the...
Are new to the S language and the S-PLUS GUI, and you want an introduction to importing data, producing simple graphs, applying statistical models, and viewing data in Microsoft Excel [®] .	<i>Getting Started Guide</i>
Are a system administrator or a licensed user and you need guidance licensing your copy of S-PLUS and/or any S-PLUS module.	<i>S-PLUS licensing Web site</i> <i>keys.insightful.com/</i>
Are a new S-PLUS user and need how to use S-PLUS, primarily through the GUI.	<i>User's Guide</i>
Are familiar with the S language and S-PLUS, and you want to use the S-PLUS plug-in, or customization, of the Eclipse Integrated Development Environment (IDE).	<i>S-PLUS Workbench User's Guide</i>
Have used the S language and S-PLUS, and you want to know how to write, debug, and program functions from the Commands window.	<i>Programmer's Guide</i>

S-PLUS documentation. (Continued)

Information you need if you...	See the...
Are familiar with the S language and S-PLUS, and you want to extend its functionality in your own application or within S-PLUS.	<i>Application Developer's Guide</i>
Are familiar with the S language and S-PLUS, and you are looking for information about creating or editing graphics, either from a Commands window or the Windows GUI, or using S-PLUS-supported graphics devices.	<i>Guide to Graphics</i>
Are familiar with the S language and S-PLUS, and you want to use the Big Data library to import and manipulate very large data sets.	<i>Big Data User's Guide</i>
Want to download or create S-PLUS packages for submission to the Comprehensive S Archival Network (CSAN) site, and need to know the steps.	<i>Guide to Packages</i>
Are looking for categorized information about individual S-PLUS functions.	<i>Function Guide</i>
If you are familiar with the S language and S-PLUS, and you need a reference for the range of statistical modelling and analysis techniques in S-PLUS. Volume 1 includes information on specifying models in S-PLUS, on probability, on estimation and inference, on regression and smoothing, and on analysis of variance.	<i>Guide to Statistics, Vol. 1</i>
If you are familiar with the S language and S-PLUS, and you need a reference for the range of statistical modelling and analysis techniques in S-PLUS. Volume 2 includes information on multivariate techniques, time series analysis, survival analysis, resampling techniques, and mathematical computing in S-PLUS.	<i>Guide to Statistics, Vol. 2</i>

CONTENTS

S-PLUS Books	iv
Chapter 1 The S-PLUS Workbench	1
Introduction	2
Terms and Concepts	3
Finding Help for the Workbench	6
Starting the S-PLUS Workbench	9
Examining S-PLUS Preferences	12
Examining the S-PLUS Workbench GUI	23
Commonly-Used Features in Eclipse	48
Chapter 2 The S-PLUS Perspective	51
Introduction	52
S-PLUS Perspective Views	53
Chapter 3 S-PLUS Workbench Debug Perspective	65
Introduction	66
Debug Perspective Options and Preferences	68
Debug Perspective Views	74
Chapter 4 S-PLUS Workbench Tasks	99
Introduction	100
S-PLUS Workbench Projects	101

Customized Perspective Views	120
Working Projects and Databases	123
S-PLUS Project Files and Views	128
S-PLUS Workbench Debugger Tasks	139
Chapter 5 Troubleshooting	153
Introduction	154
“Workspace in Use” Error	155
Working with Calls to S-PLUS GUI Functions	156
View is not visible	157
Debugging Using the Run Button	158
Subclipse Add-in error with Workbench	159
Index	161

THE S-PLUS WORKBENCH

1

Introduction	2
Terms and Concepts	3
Finding Help for the Workbench	6
Getting Started Tutorial	6
Help for S-PLUS Functions	7
The S-PLUS Workbench PDF	8
Starting the S-PLUS Workbench	9
From Microsoft Windows	9
From Unix	10
The S-PLUS Workspace	10
Examining S-PLUS Preferences	12
File Associations	12
S-PLUS Workbench options	14
Console Options	16
Editor	17
Outline Options	20
Output Options	21
Task Options	22
Examining the S-PLUS Workbench GUI	23
S-PLUS New Project Wizard	23
Customized Menus, Toolbars, and Dialogs	23
S-PLUS Workbench Status Bar	29
S-PLUS Workbench Perspectives and Views	30
Default Shared Views	37
Commonly-Used Features in Eclipse	48

INTRODUCTION

S-PLUS provides a plug-in, or customization, of the Eclipse Integrated Development Environment (IDE) called the S-PLUS Workbench. You can use the S-PLUS Workbench, the basic Eclipse IDE features, and other third-party plug-ins for many tasks, including:

- Manage your project files and tasks.
- Edit your code.
- Run S-PLUS commands.
- Examine S-PLUS objects.
- Debug your code.
- Track resource use, functions, variables, and expressions.
- Troubleshoot problems with S-PLUS code.
- Provide source control for shared project files.

The S-PLUS Workbench is a stand-alone application that runs the S-PLUS engine. When you run the S-PLUS Workbench, you do not need to run any other version of S-PLUS (for example, the console or traditional Windows or Java GUI).

Caution

If you run two or more simultaneous sessions of S-PLUS (including one or more in the S-PLUS Workbench), take care to use different working directories. To use the same working directory for multiple sessions can cause conflicts, and possibly even data corruption.

This chapter introduces the S-PLUS Workbench and provides important conceptual information and definitions of terms you need to know to use the S-PLUS Workbench most effectively.

- Chapter 2 provides reference for the S-PLUS perspective.
- Chapter 3 provides reference for the Debug perspective.
- Chapter 4 provides tasks for learning to use the S-PLUS Workbench.

TERMS AND CONCEPTS

Before you start using the S-PLUS Workbench, you should understand key terms and concepts that vary from the traditional S-PLUS for Windows GUI and S-PLUS for UNIX Java GUI.

Note

If you are using the Eclipse IDE on a UNIX platform from a Windows machine using a Windows X-server software package, you might notice that Eclipse runs slowly, similar to the S-PLUS Java GUI. See the Release Notes for more information and recommendations for improving UI performance.

Note

Eclipse version 3.2 or later does not support SPARC/Motif for Solaris. If you are using a version of Solaris prior to version 10, you must install the GTK (version 2.2.4 or greater) library. For more information about finding this library, see <http://www.sun.com/software/solaris/>. (This library is included in Solaris 10.)

Table 1.1: *Important terms and concepts.*

Term	Definition
<i>Perspective</i>	<p>Defines the preferences, settings, and views for working with Eclipse projects.</p> <ul style="list-style-type: none"> • The S-PLUS perspective is conceptually equivalent to the traditional S-PLUS Windows GUI or UNIX Java GUI. Use the S-PLUS perspective as the primary perspective for interactive S-PLUS command line use. For an example of changing the perspective, see the section Customized Perspective Views on page 120. • The Debug perspective provides an integrated debugging and profiling environment, with customized views, menu options, and behavior. For more information about using the Debug perspective, see Chapter 3, S-PLUS Workbench Debug Perspective.

Table 1.1: Important terms and concepts. (Continued)

Term	Definition
<i>Workspace</i>	<p>A physical directory on your machine that manages S-PLUS Workbench resources such as projects and other options. On your machine's hard drive, the workspace directory contains a single S-PLUS .Data database and the Eclipse .metadata database. (You should never touch these resources.) This design is different from the association you notice when you work in S-PLUS in other environments. When you start the S-PLUS Workbench, you are prompted to create or identify the workspace. See the section The S-PLUS Workspace on page 10.</p>
<i>Project</i>	<p>A resource containing text files, scripts, and associated files. The S-PLUS Workbench project is used for build and version management, sharing, and resource management. Before you begin working with any files in the S-PLUS Workbench, create a project. You can create a new project by:</p> <ul style="list-style-type: none"> • Specifying a project name and allowing Eclipse to locate the project in the workspace directory, and then selecting an existing directory containing project files at an alternate location (that is, work with the files at the specified location). • Specifying a project name and selecting an existing directory containing project files. <p>Another important concept is that of the <i>working project</i>. Set a project as the working project, which changes the working directory to the project's directory in your workspace and stores data objects in the project's .Data database. See the section Setting the Working Project on page 123 for more information.</p> <p>Important: If you select an existing S-PLUS project directory for your Workbench project, you must set that project to be the <i>working project</i> to write data objects to its .Data directory. See the section Working Projects and Databases on page 123 for a detailed discussion. See the section Creating a Project on page 102.</p>

Table 1.1: *Important terms and concepts. (Continued)*

Term	Definition
<i>View</i>	A perspective's integrated window, containing menus, options, and commands, that display specific parts of your data and projects and provide tools for data manipulation. For descriptions of the S-PLUS perspective views, see the section S-PLUS Perspective Views on page 53. For descriptions of the Debug perspective views, see the section Debug view on page 76. For practice exercises working with views, see Chapter 4, S-PLUS Workbench Tasks.
<i>Editor</i>	An integrated code/text editor that includes support for syntax coloring, text formatting, and integration with the other views. Analogous to the Script Editor in the traditional S-PLUS GUI. For more information, see the section S-PLUS Workbench Script Editor on page 45. To practice using the Script Editor, see the section Editing Code in the Script Editor on page 129.

FINDING HELP FOR THE WORKBENCH

The Eclipse IDE contains extensive, in-depth documentation for its user interface. For information about basic Eclipse IDE functionality, on the menu, see the Eclipse *Workbench User Guide*.

Getting Started Tutorial

If you are not familiar with the Eclipse IDE, after you start the S-PLUS Workbench, take the first few minutes to learn the basic concepts and IDE layout by working through the basic tutorial in the *Workbench User Guide*.

To view the Eclipse Getting Started tutorial

1. From the S-PLUS Workbench main menu, click **Help ► Help Contents**.

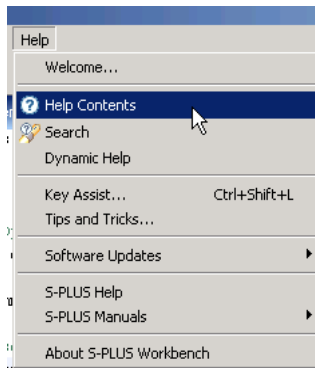


Figure 1.1: *Eclipse IDE Help menu.*

2. In the left pane, expand the table of contents by clicking **Workbench User Guide**.
3. Click **Getting Started**, and then click **Basic tutorial**.

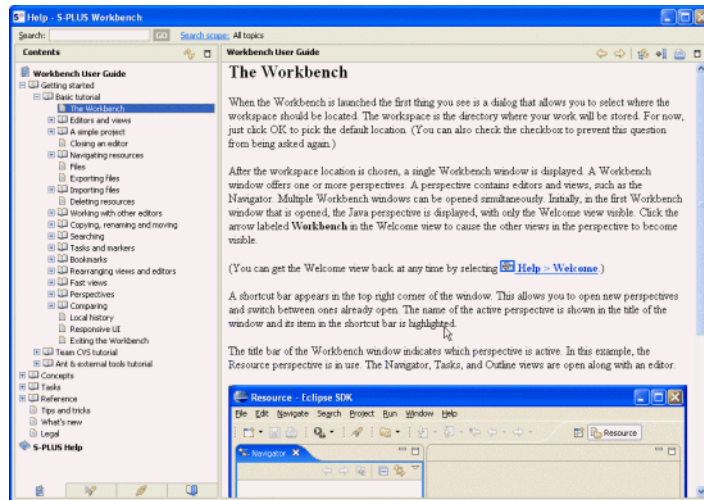


Figure 1.2: *The Eclipse basic tutorial.*

The *Workbench User Guide* opens in a separate window; you can toggle between the S-PLUS Workbench application and the Help browser.

Help for S-PLUS Functions

The S-PLUS Workbench provides access to function help topics.

- In the **Console**, type `help(functionname)` where *functionname* is the function for which you want help.
- In the Script Editor, highlight the function for which you want help, and then press F1.
- Use the S-PLUS Workbench menu options. In the Script Editor, select the function for which you want help, and then, on the menu click either:

- **S-PLUS ► Open S-PLUS Help File**

OR

- **Help ► S-PLUS Help**

The S-PLUS Workbench PDF

If you browsed to and opened this document directly from the installation directory, you might be interested to know how you can open it directly from the S-PLUS Workbench user interface.

Note

Whether you are working in Windows[®] or a UNIX[®] platform, You must have access to a PDF reader to open any of the PDFs shipped with S-PLUS.

On the S-PLUS Workbench menu, click **Help ► S-PLUS Manuals ► S-PLUS Workbench Guide**. (Note that all S-PLUS manuals are available from the **S-PLUS Manuals** menu, including the *Programmer's Guide*, the *Application Developer's Guide*, the *Function Guide*, the *Big Data User's Guide*, the *Guide to Packages* and the *Guide to Graphics*, among others.)

To specify a PDF reader, on the S-PLUS Workbench menu, click **Window ► Preferences**, and then, in the **S-PLUS** page of the **Preferences** dialog, set your PDF reader's name and location.

For more information about setting preferences, see the following documentation:

- The section Examining S-PLUS Preferences on page 12.
- The section Setting the S-PLUS Workbench Preferences on page 110.
- The Eclipse *Workbench User Guide*, available from the S-PLUS Workbench menu item **Help ► Help Contents**.

Note

To get information about using the S-PLUS Packages feature with the S-PLUS Workbench, see the *Guide to Packages*.

STARTING THE S-PLUS WORKBENCH

The S-PLUS Workbench user interface is the same in both Microsoft Windows and UNIX platforms.

From Microsoft Windows In Microsoft Windows, click the **Start** menu ► **All Programs** ► **S-PLUS 8.0** ► **S-PLUS Workbench**.

Note

When you start the S-PLUS Workbench from the Windows **Start** menu, it uses a shortcut that starts a Java virtual machine (JVM) immediately. This shortcut is defined to run a command with the option to set memory heap size: -Xmx400m.

```
<SHOME>\eclipse\eclipse.exe ...various options... -Xmx400m
```

You can override this setting and increase the memory heap size by appending a different setting at the end of the shortcut. (For example, change -Xmx400m to -Xmx600m at the end of the command to set the memory heap size to 600mb.) See Figure 1.3 for an example.

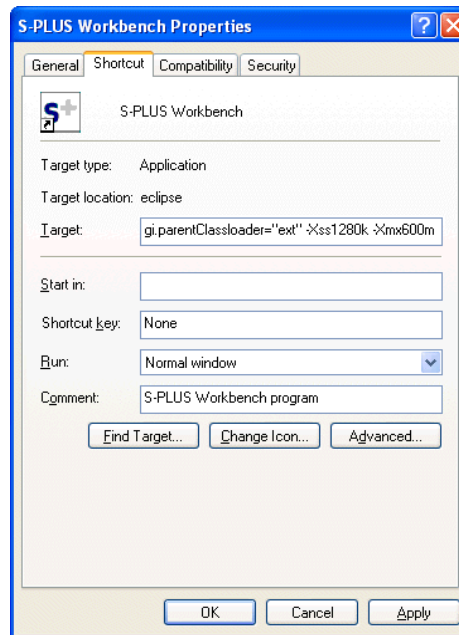


Figure 1.3: *S-PLUS Workbench Properties dialog.*

From Unix

In UNIX, at the command prompt, type

```
Splus -w
```

or type

```
Splus -workbench
```

Note

To extend the Java maximum memory heap size to 600MB, set the environment variable `JAVA_OPTIONS` to `-Xmx600m`. For example, on Solaris or Linux, start the S-PLUS Workbench with:

```
env JAVA_OPTIONS="-Xmx600m" SPlus -w
```

The S-PLUS Workspace

When you launch the S-PLUS Workbench, you see the **Workspace Launcher** dialog. You must indicate the location of the workspace.

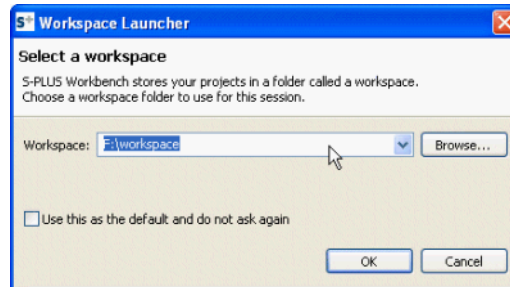


Figure 1.4: The *Workspace Launcher* dialog.

The S-PLUS workspace is the directory where the S-PLUS workspace **.Data** and Eclipse **.metadata** databases are stored. (You should never touch these files.) Optionally, the workspace directory can also store your project directories. The S-PLUS workspace is the default directory specified for the project's directory in the **New Project** wizard. See the section S-PLUS New Project Wizard on page 23 for more information. (For instruction on creating a workspace, see the section Setting the Workspace on page 101.)

Important

In the S-PLUS Workbench, you have two options for storing data objects:

- Using the S-PLUS Workbench model, where the S-PLUS workspace contains a **.Data** directory, not individual projects. The **.Data** directory can store objects for projects to share in the workspace.
- Using the familiar S-PLUS model, the *working S-PLUS project* stores its data objects to its **.Data** directory and replaces the first entry in the **Search Path** with the project's location. It is also the location to which relative paths are resolved.

Working projects are marked by an arrow icon, and by the cue (**working**) in the navigator:

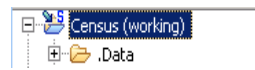


Figure 1.5: *The working project.*

For more information about setting the S-PLUS working project, see the section Setting the Working Project on page 123.

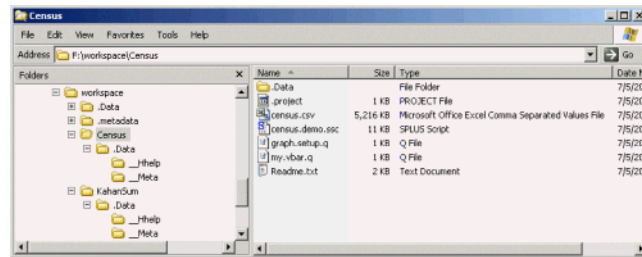


Figure 1.6: *Workspace directory (in Windows) showing .Data directory, .metadata directory, and project directories.*


Notes

When you work with S-PLUS Workbench projects, avoid nesting projects (that is, create one project in a subdirectory of another project).

To avoid conflicts, never work on S-PLUS files in the S-PLUS Workbench and another S-PLUS interface at the same time.

EXAMINING S-PLUS PREFERENCES

The S-PLUS Workbench IDE defaults are set to the S-PLUS perspective. The preferences include project type, window appearance, editor preferences, menu options, and file associations. Use the **Preferences** dialog to change these preferences and any other default Eclipse preferences. To display the **Preferences** dialog, on the main menu, click **Window ► Preferences**.

You can also display the **Preferences** dialog for the following S-PLUS Workbench views by clicking the drop-down button () and selecting **Preferences** from the control menu:

- **Tasks** view.
- **Outline** view.
- **Output** view.
- **Console** view.

You can display the **Preferences** dialog for the S-PLUS Workbench Script editor from the right-click menu (that is, right-click the Script Editor, and from the menu, click **Preferences**).

Hint

The Eclipse *Workbench User Guide* includes descriptions of the Eclipse options in the **Preferences** dialog.

For instruction on setting S-PLUS preferences, see the section Setting the S-PLUS Workbench Preferences on page 110.

The S-PLUS Workbench sets defaults for the following preferences.

File Associations

S-PLUS recognized file types include *.q, *.r, *.ssc, and *.t. Any of these files, associated with the S-PLUS Script editor, are checked for syntax errors and scanned for task tags.

Note that when you select the file type, its associated editors are displayed in the **Associated editors** box. You can add or remove both file types and associated editors.

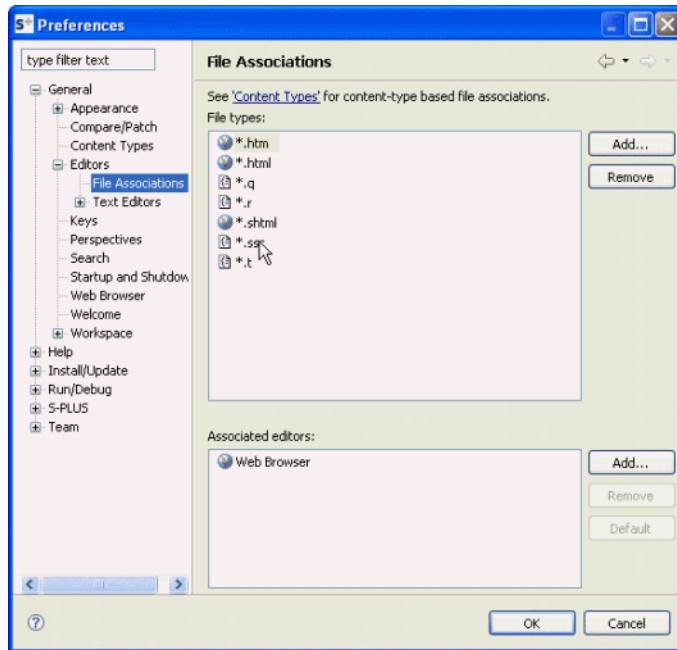


Figure 1.7: The *File Associations* page of the *Preferences* dialog.

S-PLUS Workbench options

These options control general settings for the S-PLUS Workbench.

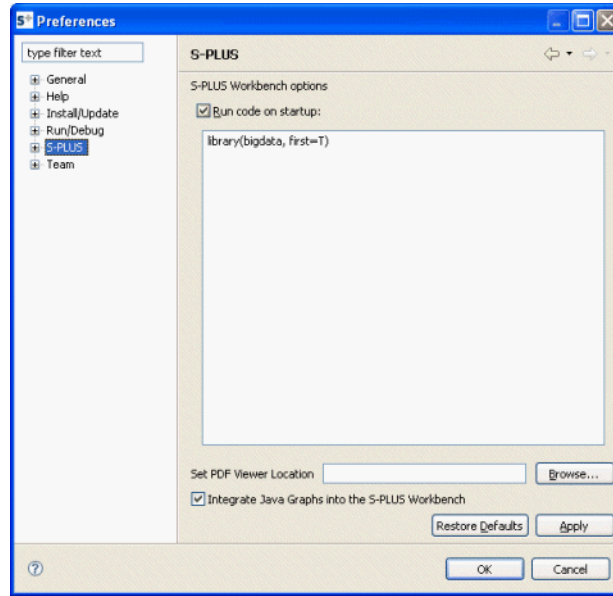


Figure 1.8: The *S-PLUS Workbench Options* page of the *Preferences* dialog.

Run code on startup

Select this option, and then provide any code that you want the S-PLUS Workbench to run when it starts up. Note that this box is selected by default, and the Big Data library is loaded by default.

Note

If you clear the **Run code on startup** box, or if you remove the option to load the Big Data library on startup, and then later open a project that uses the Big Data library, you could see unexpected results when you try to perform actions. If your typical projects include large data sets, select this option to always load the Big Data library when you start the S-PLUS Workbench.

Set PDF Viewer Location

Provide the name and path to your PDF viewer. This is used to open documents from the **Help** menu. (Most S-PLUS documentation is provided in PDF format, so you must have a PDF viewer to read the S-PLUS documentation.) If you leave this box blank, the S-PLUS Workbench attempts to use the default PDF file viewer on your system, if one is available.

Integrate Java Graphs into the S-PLUS Workbench

This option is selected by default. Clear this option if you do not want Java graphs embedded in the S-PLUS Workbench.

Note

`java.graph` is the default device for the S-PLUS Workbench.

With this option selected, any Java graphs created as part of your script appear embedded in a view to the right of the folder containing the console view by default.

Figure 1.9 shows the a Java graph from the Census sample, embedded in the S-PLUS Workbench.

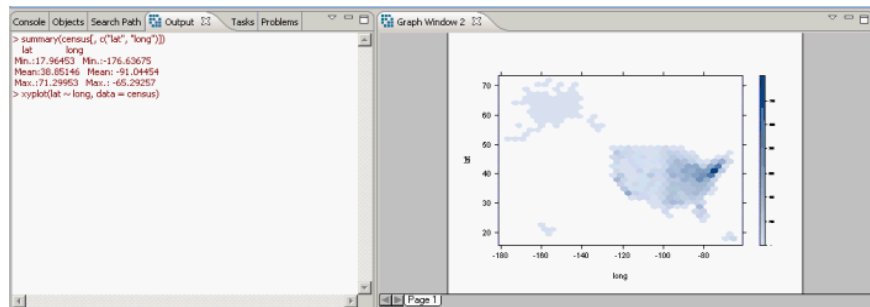


Figure 1.9: *Java graph embedded in the S-PLUS Workbench.*

Console Options

The **Console** page controls settings for the S-PLUS Workbench **Console**.

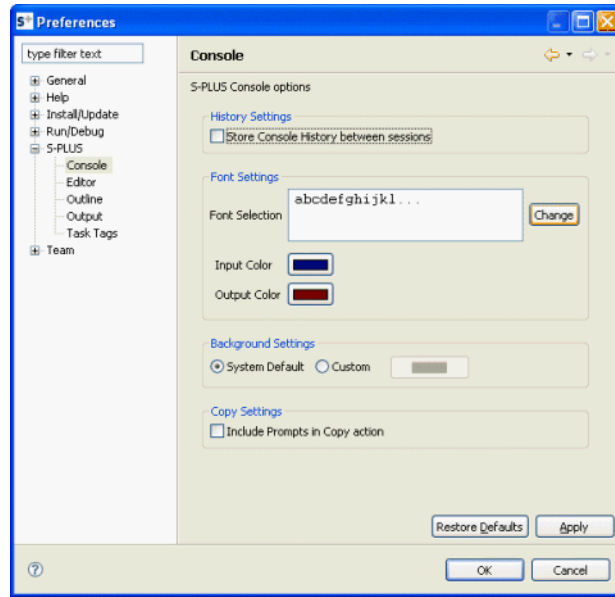


Figure 1.10: *Console* page of the *Preferences* dialog.

Store Console History Between Sessions

By default, this option is selected. It persists the commands you issue in the **Console** (which then appear in the **History**), between sessions. When you re-start the S-PLUS Workbench, click **History** to display the stored entries. Entries you select in the **History** then appear in the **Console**. Also, you can scroll up and down in the **History** to display items in the **Console**. For more information about using the **History**, see the section Examining the History view on page 136. For information about setting options for the **Output** view, see section Output Options on page 21.

Font Settings

By default, the **Console** displays input and output text using the default system font as blue and red, respectively. You can change both the font and the color.

- To set the font, click **Change**, and then, in the **Font** dialog, select from **Font**, **Font style**, **Size**, and any additional font properties to use. Note that the font changes for both input and the output displayed in the **Console**.

- To set a custom font color, click the **Input Color** or **Output Color** button, and then, in the **Color** picker, select a color for the input or output.

Background Settings

By default, the S-PLUS **Console** uses the system default. Select **Custom Color**, and then click the color button to display the **Color** picker and choose a different background color.

Include Prompts in Copy action

Select if you want to include prompts (> and +) when you copy code from either the **Console**.

Editor

These options control settings for the S-PLUS Workbench Script Editor.

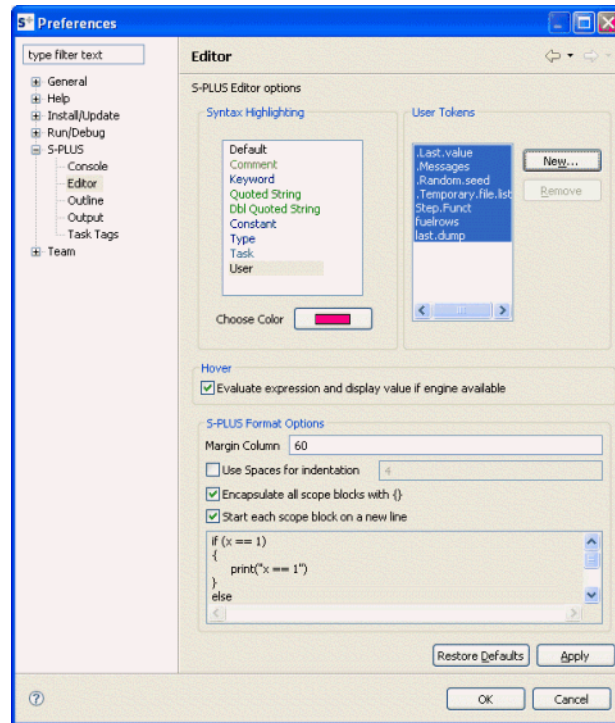


Figure 1.11: The *Editor* page of the *Preferences* dialog.

Syntax Highlighting Specifies the colors for text and defined syntax appearing in the Script Editor. To change the default color for any of the items listed, click **Choose Color** and, from the color picker dialog, select a color.

Note

To set background color, in the **Preferences** dialog, select **General ► Editors ► Text Editors**, and, in the **Appearance color options** box, select **Background color**. See the *Workbench User Guide* for more information about setting general options.

User Tokens

Lists items specified for user-defined syntax highlighting.

By default, no user-defined highlighted terms are defined. Any term you define using this option appears in the S-PLUS Script Editor in the color you define in **Syntax Highlighting** for the option **User**. To add a user-defined token, click **New**, and then, in the **Add Desired S-PLUS Text** dialog, provide the term or source.

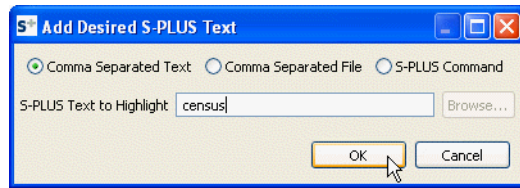


Figure 1.12: *Add Desired S-PLUS Text* dialog.

In the **Add Desired S-PLUS Text** dialog, you can provide:

- Individual terms, separated by commas.
- The contents of a comma-separated file.
- The results of an S-PLUS command. (Note that Figure 1.11 shows the results of the S-PLUS command `objects()`, which adds all objects in the current working project to the **User Token** list.

For more information about adding user tokens, see the section **S-PLUS View Preferences** on page 111.

Hover

Displays a tooltip when the mouse hovers over an expression. The tooltip displays the value of the expression, if the engine is available.

S-PLUS Format Options

Provides control over the S-PLUS Workbench's automatic code layout and formatting style.

Note

Changes you make to the **S-PLUS Format Options** do not affect your code until you select from the menu **S-PLUS ► Format**.

Table 1.2: *S-PLUS Format Options.*

Format Option	Description
Margin Column	Sets the right-hand margin to the specified character column, counting from the left-most character. By default, set to 60, making the editing space 59 characters wide.
Use Spaces for Indentation	By default, cleared. If selected, the default value is 4. If you leave this cleared, the auto-formatting feature uses tab indents, rather than character spaces.
Encapsulate all scope blocks with {}	Select to enclose all of your scope blocks with curly brackets ({}). Selected by default.
Start each scope block on a new line	Inserts a line break before the first line of a scope block.

The read-only text box appearing at the bottom of the **S-PLUS Format Options** area provides an preview of your choices.

Outline Options

Lists the options to display anonymous functions and functions to watch.

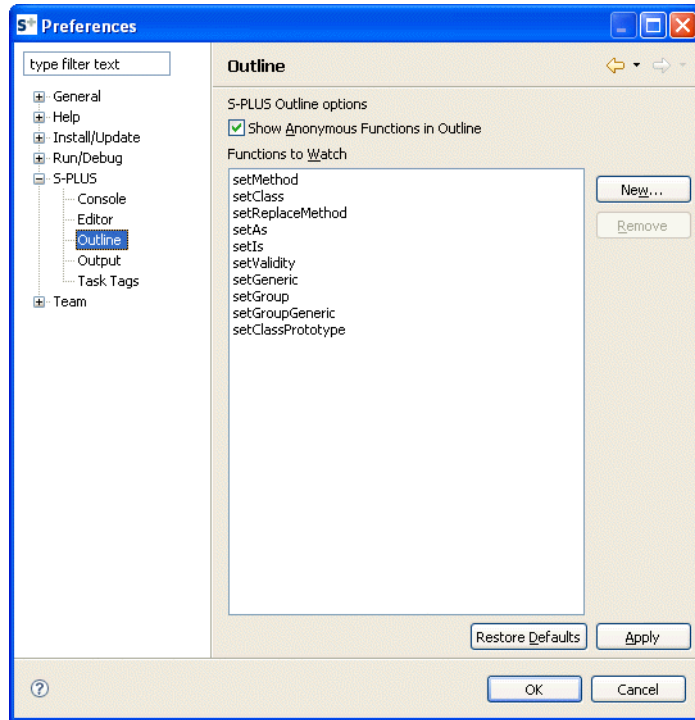


Figure 1.13: The *Outline* page of the *Preferences* dialog.

Show Anonymous Functions in Outline

By default, the S-PLUS Script Editor shows anonymous functions in the outline.

Functions to Watch

Contains a predefined list of S-PLUS functions to identify in the **Outline** view. You can add your own functions to this list using the **New** button. You can also remove functions from the list or reorder the list.

Output Options

The **Output** page controls settings for the S-PLUS Workbench **Output** view.

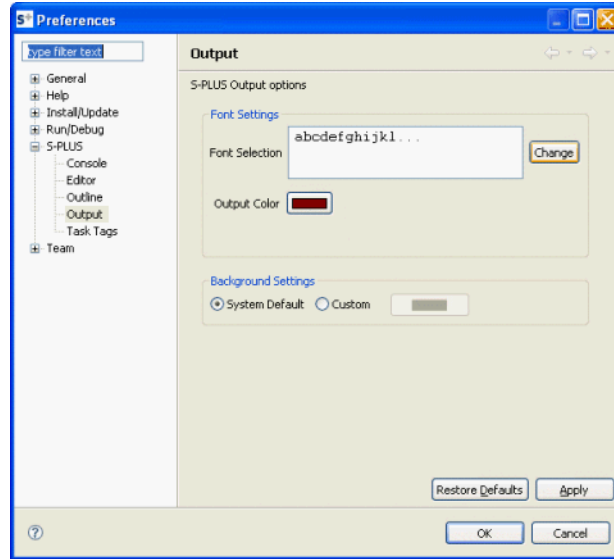


Figure 1.14: *Output* page of the *Preferences* dialog.

Font Settings

By default, the **Output** view displays output text using the default system font as red. You can change both the font and the color.

- To set the font, click **Change**, and then, in the **Font** dialog, select from **Font**, **Font style**, **Size**, and any additional font properties.
- To set a custom font color, click the **Output Color** button, and then, in the **Color** picker, select a color for the output.

Background Settings

By default, the **Output** view uses the system default. Select **Custom Color**, and then click the color button to display the **Color** picker and choose a different background color.

Task Options Lists the three pre-defined default task tags. See the section Tasks view on page 61 for more information.

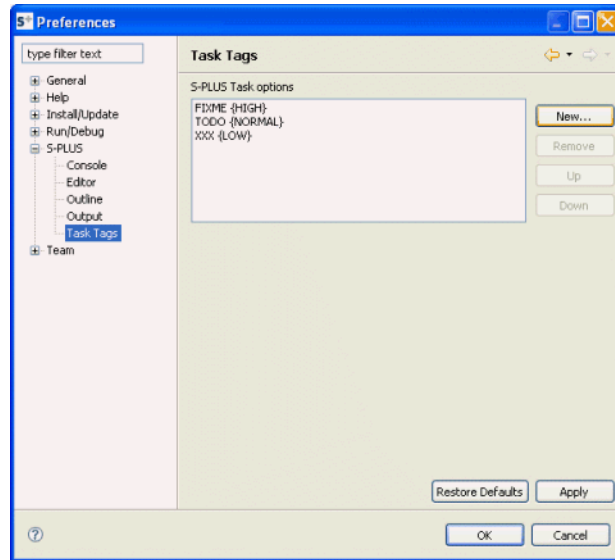


Figure 1.15: The *Task Tags* page of the *Preferences* dialog.

EXAMINING THE S-PLUS WORKBENCH GUI

After the S-PLUS Workbench GUI opens, and you set preferences, spend a moment examining the user interface, including the toolbars, menus, perspectives, and views.

- For more information about perspectives, see the section S-PLUS Workbench Perspectives and Views on page 30.
- For more information about views, see the section Examining the Views on page 32.

S-PLUS New Project Wizard

When you start a new S-PLUS project in the S-PLUS Workbench, you see the **New Project** wizard, where you specify the location of your project files. See the section Creating a Project on page 102 for more information about specifying the project file location.

Working with Files External to the Project

You can use the Eclipse editor to edit non-project files in the S-PLUS Workbench. To open a non-project file, on the **File** menu, click **Open File**, and then browse to the location of the file to edit. For more information about editing files in Eclipse, see the *Eclipse Workbench User Guide*.

Customized Menus, Toolbars, and Dialogs

The S-PLUS Workbench includes in the Eclipse GUI:

- Customized top-level menu items.
- Customized top-level toolbar.
- Customized view-specific toolbars and view menus. (See the section Control and Right-Click Menus on page 34 for more information about the menus.)

Customized menus

S-PLUS customizes the basic Eclipse menu to provide easy access to global S-PLUS control and to control debugging options.

S-PLUS Menu

The **S-PLUS** menu contains the following options:

Table 1.3: *S-PLUS menu options.*



S-PLUS Menu Option	Description
Format	Applies S-PLUS consistent formatting and line indentation to the entire script.
Toggle Comment	Designates the selected text in the Script editor as a comment, or, if the selected text already is a comment, removes the comment designation
Shift Right	Moves the selected text to the right.
Shift Left	Moves the selected text to the left.
Define Folding Region	<p>In the S-PLUS Script Editor, sets the currently-selected code block as a collapsible block. A collapsible region is indicated by the icon  in the left margin, and a vertical line, marking the region to collapse. A collapsed region is indicated by the icon  .</p> <p>In Windows, you can hover the mouse pointer over the collapsed region to display the contents of the region in a tooltip.</p>
Run Selection	<p>Runs the code that is currently selected. If nothing is selected, the current editor contents are run.</p> <p>This menu item also appears in the right-click menu of the S-PLUS Script Editor and is represented by the Run S-PLUS Code button on the main S-PLUS Workbench toolbar.</p>
Run Current File	Runs the entire contents of the script currently open in the S-PLUS Script Editor.

Table 1.3: S-PLUS menu options. (Continued)

S-PLUS Menu Option	Description
Find “function”	<p>Finds the selected function definition and opens it for editing.</p> <p>Find looks first in files currently open in an editor, then it looks through your workspace. Finally, it searches the S-PLUS database.</p> <p>If the function is not found in an editor and multiple definitions exist in the workspace, use the resulting dialog to indicate the proper source.</p> <p>Note: Highlighting a function and typing CTRL+mouse click also opens the selected function definition for editing.</p> <p>See the section To edit a function definition on page 130 for more information</p>
Find References	<p>Locates and highlights all instances of a function call in a workspace. Find References opens the Search view and displays the number of times in a workspace where the selected function is called. You can See the Eclipse <i>Workbench User Guide</i> for more information about the Search view.</p> <p>See the section To find all references to a function on page 131 for more information.</p>
Copy to Console	<p>Copies the selected code and pastes it into the Console view. See the section Copying Script Code to the Console on page 135</p>
Open S-PLUS Help File	<p>Opens the S-PLUS Language Reference to the topic for the selected function. If you have no documented function selected, the help function topic is displayed.</p>

Run Menu

The **Run** menu varies, depending on the perspective selected. In both the S-PLUS and Debug perspective, the following S-PLUS Workbench options are available. (See the corresponding descriptions for the toolbar buttons in the section The S-PLUS Workbench Toolbar on page 27 for more information.)

- **Run S-PLUS Code**
- **Run Next S-PLUS Command**
- **Stop S-PLUS Code**
- **Toggle S-PLUS Debugger**
- **Toggle S-PLUS Profiler**
- **Toggle S-PLUS Warning Breakpoint**
- **Toggle S-PLUS Error Breakpoint**

For more information about the **Run** menu options available only in the Debug perspective, see the section Debug Run Menu Options on page 70.

Window Menu

The S-PLUS Workbench preferences are available from the **Window ► Preferences** menu option. See the section Examining S-PLUS Preferences on page 12 for more information.

Help Menu

Reference help, conceptual help, books, and user-interface guidance are available from the Help menu.

- Click **S-PLUS Help** from the **Help** menu to display the S-PLUS Language Reference topic for the help function.
- Click **S-PLUS Manuals** for a list of the PDFs that are installed by default with your S-PLUS installation.

Customized Toolbars

Both S-PLUS perspectives in the Workbench provide customizations to the Eclipse toolbar and to view-specific toolbars.

The S-PLUS Workbench Toolbar

Regardless of the displayed perspective, the S-PLUS Workbench toolbar appears in the IDE.

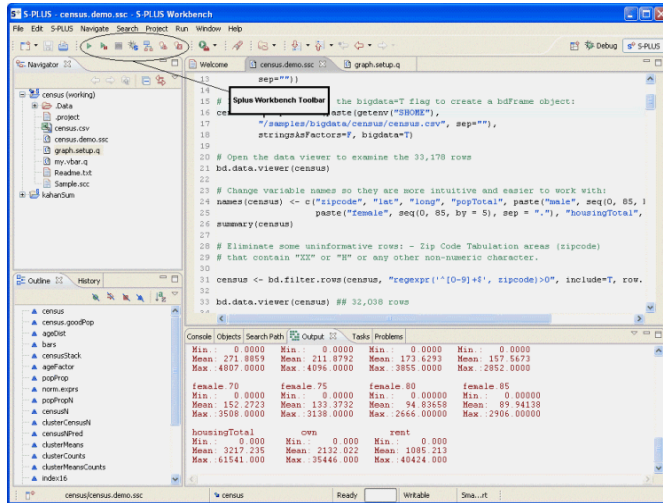


Figure 1.16: The S-PLUS Workbench toolbar.

Note

Eclipse implements a **Run** menu item that is different from that of S-PLUS Workbench implementation. Use the S-PLUS Workbench **Run** menu item.

Use the S-PLUS Workbench toolbar to control running, debugging, breaking, and profiling your code.

Table 1.4: S-PLUS Workbench toolbar.


Button	Description
	Run S-PLUS Code. Click in either the Debug or the S-PLUS perspective to run code that appears in the editor. (To view the output, select the Output .)

Table 1.4: *S-PLUS Workbench toolbar. (Continued)*







Button	Description
	<p>Run Next S-PLUS Command. Looks for the current selection and runs the top-level S expression found at that location.</p> <p>If the cursor location does not match an expression exactly, the next expression is evaluated (rather than the first one).</p> <p>The output is routed to the Output, and the next expression is selected automatically (or the first expression in the script is selected automatically, if the expression that was just run was the last one).</p>
	<p>Stop S-PLUS Code. Click in either the Debug or the S-PLUS perspective to stop running code.</p>
	<p>Toggle S-PLUS Debugger. Engages the S-PLUS debugger. (You can engage the S-PLUS debugger in either the S-PLUS or the Debug perspectives; however, by default, the views displaying debugging information are visible in the Debug perspective.)</p> <p>After you engage the S-PLUS debugger, any expression you type in the Console, or that you run by clicking Run S-PLUS Code on the toolbar, invokes the S-PLUS debugger.</p>
	<p>Toggle S-PLUS Profiler. Engages the S-PLUS Profiler. (You can engage the S-PLUS Profiler in either the S-PLUS or the Debug perspectives; however, by default, the views displaying profiling information are visible in the Debug perspective.)</p> <p>You do not need to engage the S-PLUS debugger in order to engage the Profiler. See the section Profiler on page 96 for more information.</p>
	<p>Toggle S-PLUS Warning Breakpoint. Requires that the S-PLUS debugger be toggled on. Stops execution if S-PLUS encounters a warning. See Table 3.7 in the section Breakpoints view on page 89 for more information about warning breakpoints.</p>

Table 1.4: *S-PLUS Workbench toolbar. (Continued)*

Button	Description
	Toggle S-PLUS Error Breakpoint. Requires that the S-PLUS debugger be toggled on. Stops execution if S-PLUS encounters an error. See Table 3.7 in the section Breakpoints view on page 89 for more information about error breakpoints.

View toolbars

For more information about individual views' toolbars, see the individual views' descriptions. See the section Examining the Views on page 32 for more information.

S-PLUS Workbench Status Bar

The Workbench features a status bar that provides important information about the working project and the current view.

**Figure 1.17:** *Status bar.***Table 1.5:** *S-PLUS Workbench status bar.*


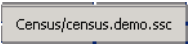
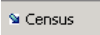


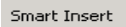

Status Item	Description
	<p>Show View as Fast View. An Eclipse feature.</p> <ul style="list-style-type: none"> Click to display a list of available views, and then select a view to maximize it and add its icon to the status bar. Click the view's icon in the status bar to minimize the view. (Alternatively, click the view's minimize icon, in its upper right corner, to minimize it.) <p>For more information, see the Eclipse <i>Workbench User Guide</i>.</p>
	<p>Current working directory and file. When the Script Editor has focus, this section of the status bar displays the current file.</p>

Table 1.5: *S-PLUS Workbench status bar. (Continued)*

Status Item	Description
	Working project. Displays the name of the project that is currently set as the <i>working project</i> . For more information about the working project, see the section Working Projects and Databases on page 123.
	Status indicator. When the box is labeled Busy , and the status indicator is filling, then code is currently running. When the box is clear and reads Ready, no code is running.
	File attribute. Indicates whether the file is read-only or writable.
	Smart Insert. Toggles the insert mode. To toggle this view, type CTRL+SHIFT+INSERT. When Smart Insert mode is toggled off, typing aids like automatic indentation, closing of brackets, and so on, are not available. Smart Insert is an Eclipse feature.
	Cursor position. Indicates the line and column position of the cursor.

S-PLUS Workbench Perspectives and Views

The S-PLUS Workbench plug-in for Eclipse includes two customized perspectives:

- The S-PLUS perspective
- The Debug perspective.

(See Table 1.1 for a short description of the perspectives.) By default, each perspective includes Eclipse views and customized S-PLUS Workbench views.

Changing the S-PLUS Workbench Perspective

You can change the perspective to suit your development style by moving, hiding, or closing views. For more information about customizing the views within the perspective, see the section Customizing the S-PLUS Perspective Views on page 54. For practice exercises customizing the perspective, see the section Customized Perspective Views on page 120.

- To customize the default S-PLUS perspective, on the menu, click **Window ► Customize Perspective**. The **Customize Perspective** dialog has two pages: **Shortcuts** and **Commands**. Each of these pages describes global changes you can make to the perspective.
- To save a changed perspective, click **Window ► Save Perspective As**.
- To restore an unsaved perspective's default settings, click **Window ► Reset Perspective**.
- To open another perspective, click **Window ► Open Perspective**, and then select a perspective from the **Select Perspective** dialog.

Figure 1.18 shows the S-PLUS perspective with the views set at their default positions.

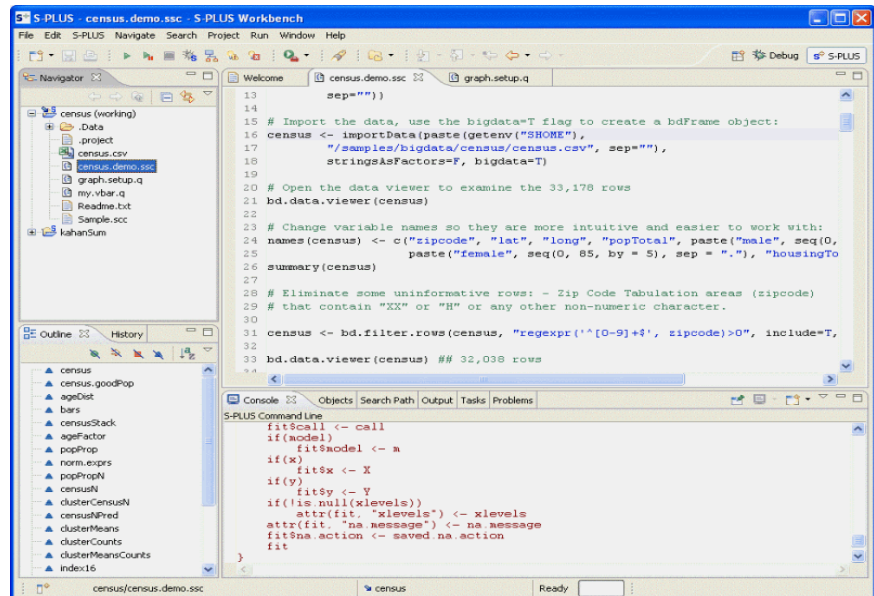


Figure 1.18: S-PLUS Workbench window, S-PLUS perspective.

Examining the Views

A view is a visual component in the workbench. Views support the script editor by providing alternate means of navigating through, working with, and examining the elements of the project.

Using the standard Eclipse IDE features, you can:

- Close a view by clicking the **X** icon on the view tab.
- Reposition a view by clicking its tab and dragging it to another part of the UI.
- Set a selected view to “Fast View.” This option hides the view to free space in the **Workbench** window and places a minimized icon, which you can click to open the view, on the status bar.
- Change the views you see in the perspective. See the section *To change the displayed views* on page 121.

Most views have their own control menus. (See the section *Control and Right-Click Menus* on page 34 for more information.)

Saving views items changed in views

When you modify an item in a view, it is saved immediately. Normally, only one instance of a particular type of view can exist in the Workbench window.

Perspective views

The following table lists the views shown by default in each perspective and indicates which views are shared by both perspectives. This section includes descriptions for the views shared across S-PLUS Workbench perspectives.

Table 1.6: *Default views in the S-PLUS Workbench perspectives.*

View Name	S-PLUS Workbench Perspective	Debug Perspective	Description reference
Allocations view		x	page 98
Breakpoints view		x	page 89
Console view	x	x	page 37
Debug view		x	page 76
Expressions view		x	page 87
Function Calls view		x	page 97
History view	x		page 54
Navigator	x	x	<i>Eclipse Workbench User Guide</i>
Objects view	x		page 56
Outline view	x	x	page 41
Output view	x	x	page 43
Problems view	x		page 59


Table 1.6: *Default views in the S-PLUS Workbench perspectives. (Continued)*

View Name	S-PLUS Workbench Perspective	Debug Perspective	Description reference
Script Editor	x	x	page 44
Search Path view	x		page 60
Tasks view	x	x	page 61
Variables view		x	page 82

Hint

Change the view layout by moving views around the IDE, or control the views displayed using the **Show View** dialog. For more information, see the section To change the displayed views on page 121.

Control and Right-Click Menus

Views contain their own control and/or right-click menus, with menu items that act on the view display or on the type of data displayed in the view. Menus are displayed either when you click the drop-down button () , located in the upper right corner of each view, or when you right click the body of the view.

The following two images show the two types of menus in the **Navigator** view. For more information about the **Navigator**, see the section Navigator on page 39.

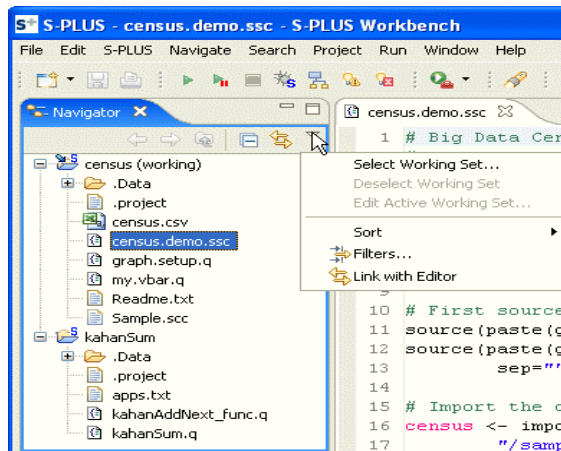


Figure 1.19: Control menu, available via drop-down button.

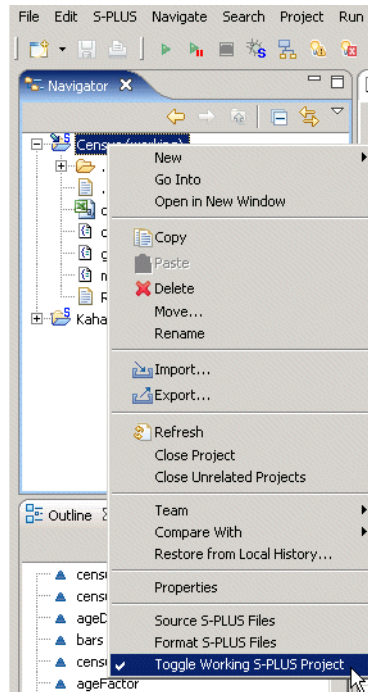


Figure 1.20: Context-sensitive menu, available via right-click in the view.

- The Script Editor has only a context-sensitive menu, available via a right-click action. Its available options depend on the current selection in the editor. For example, if you select text and right-click, you have the option to cut or copy the selection. If you select an S-PLUS function, you have the option to open the S-PLUS Help file for that function.

The options on the Script Editor's context-sensitive menu are a selection of options that appear on the main menu.

- The **Outline** view and the **Allocations** view have only the control drop-down menu.

In the following views, the right-click (context-sensitive) menu and the control drop-down menu are identical. The control menu for each view is described in this document in the section describing its view. See the section Examining the Views on page 32 for more information:

- **Console** view.
- **Function Calls** view.
- **History** view.
- **Objects** view.
- **Output** view.
- **Search Path** view.

The following views each contain two different menus:

- The control menu, available from the drop-down button.
- The context-sensitive menu, available via right-click in the body of the view. The options available on the right-click menu vary according to the item selected in the view (for

example, removing a selection, copying a selection, and so on.) For more information about where each view appears, see the section Perspective views on page 33.

Table 1.7: *Views with different control and right-click menus.*

View	Location of more menu information
Navigator	page 39
Tasks view	page 63
Problems view	page 59
Debug view	page 78
Variables view	page 84
Expressions view	page 84
Breakpoints view	page 89

Each view action also has a quick-key sequence to perform an action. (For example, to clear the text in the console, with the **Console** active, type CTRL+L.)

Default Shared Views

The following sections describe the views that are shown, by default, in both the S-PLUS perspective and the Debug perspective.

S-PLUS Workbench Console

The S-PLUS Workbench **Console** is an editable view, analogous to the **Commands** window in the S-PLUS GUI. Using the **Console**, you can:

- Run individual S-PLUS commands by typing them and pressing ENTER.
- Scroll through previous commands by pressing the UP or DOWN arrow on the keyboard.

- Copy an individual command or blocks of commands from the Script Editor, using the **Copy to Console** menu item, to run them in the **Console**. (Note that you do not need to select **Paste**; **Copy to Console** copies your selected text in the Script Editor and pastes it into the **Console**.)

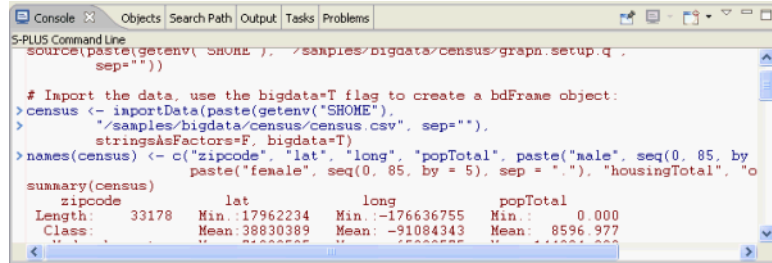


Figure 1.21: S-PLUS Workbench **Console**.

- Copy from the console to a script file. (You can also copy the command prompts. To set this option, on the menu, click **Window ► Preferences**, and on the **Console/Output** page, select **Include Prompts in Copy action**.)

Console control and right-click menus

The **Console** controls include these Eclipse options:

- Pin the view in place ().
- Toggle between open **Console** views ().
- Open a new **Console** ().


The **Console** control menu and right-click menu are the same. You can use the **Console** control menu (click or right-click the body of the **Console**) to perform the following tasks:

- Clear the contents of the console.
- Copy the selected text.
- Cut the selected text.
- Paste text from the clipboard to the console.
- Find a string.
- Select all text.
- Save the console contents to a file.

- Print the console contents.
- Open the **Preferences** dialog to set such options as font color and style, among others.

For exercises using the S-PLUS Workbench **Console**, see the section Copying Script Code to the Console on page 135. For more information about the S-PLUS **Commands** window, see Chapter 10, Using the Commands Window in the *S-PLUS User's Guide*.

Navigator

The **Navigator** is a standard Eclipse view. Its drop-down () control menu is standard to Eclipse, while its right-click menu contains three S-PLUS-specific items, described in the following section. For information about using the **Navigator**, see the Eclipse *Workbench User Guide*, available from the **Help ► Help Contents** menu.

Navigator control and right-click menus

In addition to having a drop-down control menu, the **Navigator** has a right-click menu containing three S-PLUS-specific options:

Table 1.8: *Navigator S-PLUS-specific right-click menu options.*

Menu option	Description
Source S-PLUS Files	Parses and then evaluates each expression in the selected project or file. (Note that if you have selected the project, every file in that project is sourced.)
Format S-PLUS Files	Applies S-PLUS consistent formatting and line indentation to all scripts in the working project. To customize the formatting options, on the main menu, click Window ► Preferences , in the left pane, click S-PLUS to expand the view, and then click Editor . Use the Editor page to customize formatting options. See the section To change the code formatting options on page 117 for more instruction.

Table 1.8: *Navigator S-PLUS-specific right-click menu options. (Continued)*

Menu option	Description
Toggle Working S-PLUS Project	<p>Available when you select a project in the Navigator. The project that you set as working becomes the current working directory, or the root to which all relative paths are resolved.</p> <p>The working project also becomes the first position (in the search path, which you can see in the Search Path. This path contains the .Data database. All objects created as a result of running code in the S-PLUS Workbench are written to that .Data database (regardless of the project the code is in).</p> <p>When you toggle off (that is, clear) the selection and have no working project, the .Data database is set to the current workspace, and the Search Path shows the workspace in the first position. In this case, all objects created in any project are written to the .Data database in the workspace and are available to any project in the workspace.</p> <p>See Figure 1.22 for an illustration. For more information about working projects and the current working directory, see the section Setting the Working Project on page 123.</p>

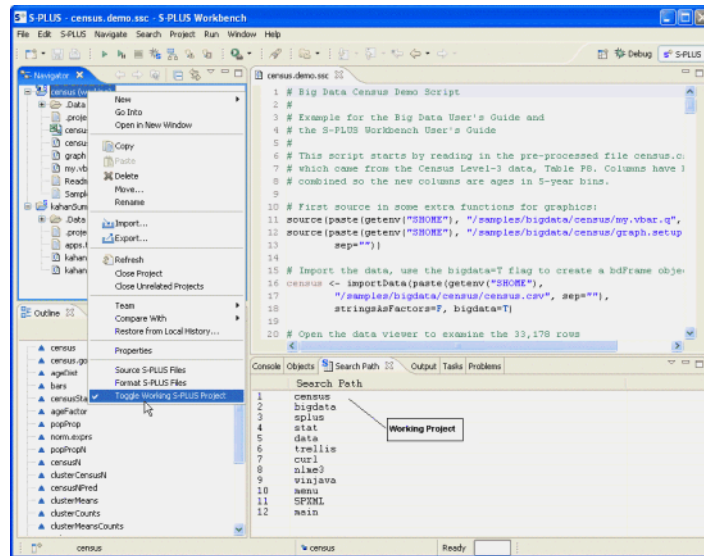


Figure 1.22: Toggle Working S-PLUS Project shows current working project at the top of the **Search Path**.

Outline view

The **Outline** view displays an outline of the elements in the script open in the script editor. In the S-PLUS Workbench, **Outline** view displays functions and objects in the order they appear in the script editor. Items that you have identified to “watch” in the **Functions to**

Watch text box of the **Preferences** dialog appear in the **Outline** view with an arrow. You can jump to the definition of a function or object (or other structure element) by clicking it in **Outline** view.

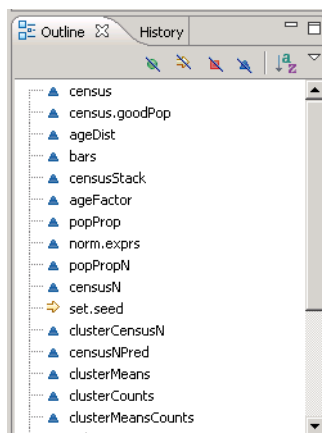


Figure 1.23: *S-PLUS Workbench Outline view.*







Note

The **Outline** view updates only after you save changes to its associated file, displayed in the Script Editor.

Outline view toolbar

The **Outline** view contains a toolbar that displays the following toggle buttons:

Table 1.9: *Outline view buttons.*

Button	Description
	Hides all standard functions displayed in the Outline view. Click again to display standard functions.
	Hides all functions that you have designated to watch displayed in the Outline view. Click again to display the functions.
	Hides all anonymous functions displayed in the Outline view. Click again to display the functions.
	Hides all variables in the Outline view. Click again to display the variables.
	Sorts items displayed the Outline view alphabetically. Click again to return the items to the order in which they appear in the script.
	Displays a menu showing all buttons available on the button bar. (You can toggle these selections either using the menu, or on the button bar.)

Outline view control menu

The **Outline** view control menu provides menu access to the buttons visible on the **Outline** view toolbar, and to the **Preferences** dialog. See the descriptions in Figure 1.9 for more information. (The **Outline** view contains no right-click menu.)

Output

The **Output** displays the code you run (and the results of the code you run) when you click either **Run S-PLUS Code** on the toolbar, or when you press F9. The text displayed in the **Output** is replaced each

time you click **Run S-PLUS Code** or press F9. That is, unlike the **Console**, the **Output** does not store and display previously-run commands. Also unlike the **Console**, the **Output** is not editable; however, you can select and copy lines of text in the **Output**. You can also print or clear the entire contents of the **Output**.

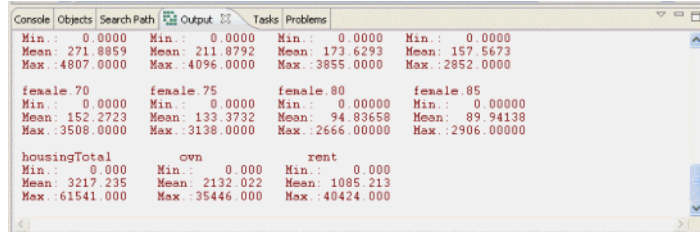



Figure 1.24: S-PLUS Workbench Output.

Output control and right-click menus

You can use the **Output** control menu (click ) to perform the following tasks:

- Clear the contents of the view.
- Copy the selected text.
- Find a string.
- Select all text.
- Save the view contents to a file.
- Print the view contents.
- Display the **Preferences** dialog to change the font color and style.

The drop-down control menu and the right-click context-sensitive menu are identical in the **Output**.

S-PLUS Workbench Script Editor

The S-PLUS Workbench Script Editor is a text editor displayed in both the S-PLUS perspective and the Debug perspective. The S-PLUS Workbench Script Editor is similar to the Script Editor in S-PLUS; however, it contains additional script-authoring features such as syntax coloring and integration with the other views in the IDE.

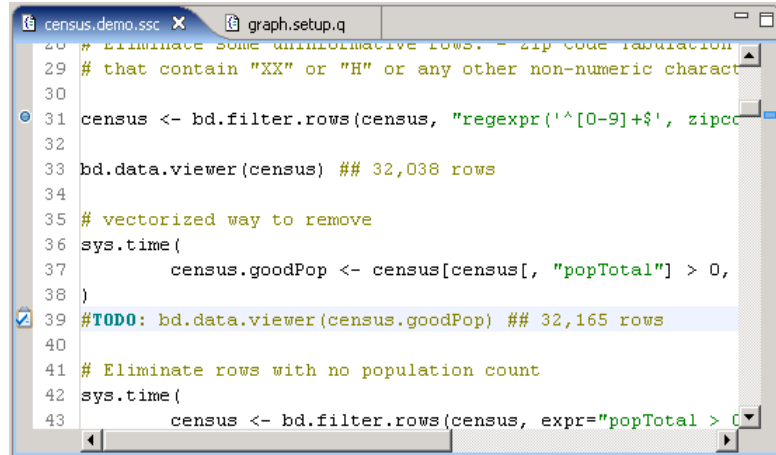


Figure 1.25: *S-PLUS Workbench Script Editor.*

You can run code in the S-PLUS Workbench Script Editor by highlighting the code and clicking **Run S-PLUS Code** (▶) on the toolbar.

Note

To interrupt code that you run from the Script Editor, either click **Stop S-PLUS Code** (on the toolbar) or press ESC.

Text Editing Assistance

To help you write efficient, easy-to-follow scripts, the Script Editor provides the following features:

- Displays keywords, user-defined text, and function arguments in customizable colors. See the section *Setting the S-PLUS Workbench Preferences* on page 110.
- Displays code line numbers in a column adjacent to the code.

- Provides automatic code indentation and parenthesis matching. (See the Eclipse documentation for more information on the editor's standard features.)

Note

To indent selected text, first highlight the text to be indented, and then press TAB or CTRL+TAB to shift the selected text right or left, respectively.

- Activates the Script **Outline** view when you edit a script.
- Displays task and breakpoint markers in the left margin, and a task marker “thumb” in the right margin.
- Displays the help topic for documented functions when you select the function name, and then type F1.

Note

You can use the Eclipse editor to edit non-project files in the S-PLUS Workbench. To open a non-project file, on the **File** menu, click **Open File**, and then browse to the location of the file to edit. For more information about editing files in Eclipse, see the Eclipse *Workbench User Guide*.

View integration

The Script Editor is closely integrated with the views in both the S-PLUS perspective and Debug perspective. This integration includes the following:

- When you type a task keyword in the editor, it is automatically added to the **Tasks** view after you save the file. See the section Tasks view on page 61 for more information.
- When you set a breakpoint, the breakpoint appears in the margin of the Script Editor both in the Debug perspective and the S-PLUS perspective. (You can also set a breakpoint in the margin of the Script Editor in both perspectives. See the section Setting breakpoints on page 141.)
- When you make an error and save your script file, the error shows in the **Problems** view. See the section To examine problems on page 137 for more information.

- When you create a new object in the script, it appears in the **Outline** view. To make it appear in the **Objects** view, you must run the script and refresh the **Objects** view.

Script Editor right-click menu

The right-click menu in the Script Editor combines actions from the Eclipse main menu, including the options available from the **S-PLUS** menu, and the **Preferences** dialog.

See the section S-PLUS Menu on page 24 for information about the S-PLUS options.

See section Setting the S-PLUS Workbench Preferences on page 110 for information about setting **Preferences** options.

(The Script Editor has no drop-down control menu.)

COMMONLY-USED FEATURES IN ECLIPSE

The core Eclipse IDE contains many additional features that you might find helpful in managing your projects. The following table lists a few of these features, along with references to the Eclipse *Workbench User Guide* to help you learn how to use them effectively.

Table 1.10: *Eclipse Tasks and Features.*

Task	Eclipse Feature Description
Comparing files with previous versions.	The Compare With Local History menu item is available from the control menu in Navigator view. Using this feature, you can compare the current version of the selected file with previously-stored local versions. For more information, see the topic “Local history” in the Eclipse <i>Workbench User Guide</i> .
Replacing files with a previous version.	The Replace With Local History and Replace With Previous from Local History menu items are available from the control menu in Navigator view. Using these features, you can replace the current version of the selected file with one of the previously-stored local versions. Replace With Previous from Local History displays no selection dialog; it just replaces the file. To choose a previous state in the Local History list, use Replace With Local History . For more information, see the topic “Replacing a resource with local history” in the Eclipse <i>Workbench User Guide</i> .
Finding a word in a project or a term in a Help topic.	Using the Search ► File menu item, you can find all occurrences of a word in a project or Help topic. For more information, see the topic “File search” in the Eclipse <i>Workbench User Guide</i> .

Table 1.10: *Eclipse Tasks and Features. (Continued)*

Task	Eclipse Feature Description
Filter files in the Navigator view.	Using the Working Sets menu option on the control menu in Navigator view, you can create subsets of files to display or hide. For more information, see the topics “Working Sets” and “Showing or hiding files in the Navigator view” in the Eclipse <i>Workbench User Guide</i> .
View a file that is not part of your project.	Use the File ► Open File menu item to open a file that is not part of your project.

THE S-PLUS PERSPECTIVE

2

Introduction	52
S-PLUS Perspective Views	53
Customizing the S-PLUS Perspective Views	54
History view	54
Objects view	56
Problems view	59
Search Path view	60
Tasks view	61

INTRODUCTION

S-PLUS Workbench perspectives define the appearance and behavior of the S-PLUS Workbench Eclipse plug-in, including the S-PLUS Script Editor, views, menus, and toolbars. The S-PLUS perspective combines S-PLUS Workbench views and options so you can accomplish specific types of tasks and work with specific types of resources.

- For more information about the S-PLUS Debugger perspective options and views, see Chapter 3.
- For practice instruction using the features in the S-PLUS perspective, as well as the S-PLUS Workbench and Debug perspective, see Chapter 4, S-PLUS Workbench Tasks.

Note

You can change a perspective to suit your development style by moving, hiding, or closing views. For more information about customizing the views within the perspective, see the section Changing the S-PLUS Workbench Perspective on page 31, or see the section Customized Perspective Views on page 120.

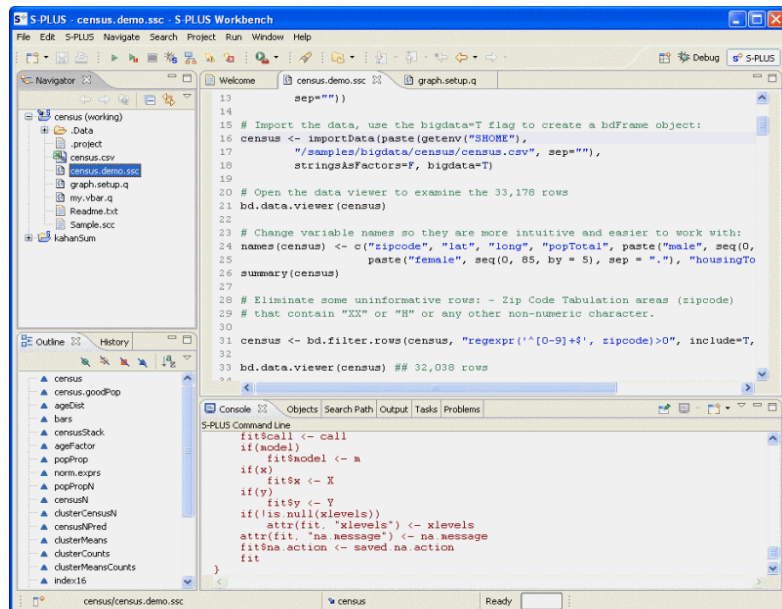


Figure 2.1: The S-PLUS perspective.

S-PLUS PERSPECTIVE VIEWS

The S-PLUS Workbench includes views shared across perspectives. For a list of all views and their default perspectives, see Table 1.6 in Chapter 1. (Chapter 1 also includes descriptions of the shared views.)

The S-PLUS perspective includes default Eclipse views and customized views. Customized views in the S-PLUS perspective include the following:

Table 2.1: *S-PLUS perspective views and exercise references.*

View	Descriptions	Practice exercises
Console view	Shared view. For a description, see the section S-PLUS Workbench Console on page 37.	"To run copied script code" on page 136
History view	For a description, see the section History view on page 54.	"To examine the history" on page 137
Objects view	For a description, see the section Objects view on page 56.	"To examine the objects" on page 132
Outline view	Shared view. For a description, see the section Outline view on page 41.	"To examine the outline" on page 131
Output view	Shared view. For a description, see the section Output on page 43.	"To run code" on page 137

Table 2.1: *S-PLUS perspective views and exercise references. (Continued)*

View	Descriptions	Practice exercises
Problems view	For a description, see the section Problems view on page 59.	"To examine problems" on page 137
Search Path view	For a description, see the section Search Path view on page 60.	"Adding a Database" on page 125 and "Detaching a Database" on page 126
Tasks view	Shared view. For a description, see the section Tasks view on page 61.	"To add a task in the script file" on page 134 and "To add a task directly to the Tasks view." on page 133

Both the S-PLUS perspective and the Debug perspective also display the default Eclipse **Navigator** view, which displays project directories and all files associated with each project. The **Navigator** view and other Eclipse IDE views are described in the Eclipse *Workbench User Guide*.

Customizing the S-PLUS Perspective Views

The default S-PLUS perspective settings control the views that open by default in preset locations; however, you can customize the view appearance, and then save the resulting perspective. See the section Customized Perspective Views on page 120 for more information.

The following sections describe only the views that appear by default in only the S-PLUS perspective.

History view

The **History** view is similar to the **Commands History** dialog in S-PLUS for Windows. The **History** view is a scrollable list of commands that have previously been run in the **Console**. (Commands that you run by clicking **Run S-PLUS Code** or pressing F9 do not appear in the **History** view. See the section Output on page 43.)

- When you select a command in the **History** view, the pending text in the **Console** changes to the selected text. You can then press ENTER, or you can double-click the text in the **History** view to execute the command. You can select only one line at a time in the **History** view.
- When you scroll up or down through previously-run commands in the **Console**, the corresponding command is highlighted in the **History** view.

Note

In Windows, S-PLUS uses the key F10 to run a selected command. The S-PLUS Workbench uses the key F9 to run a selected command in all platforms.

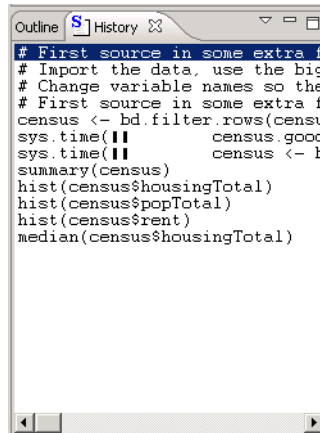


Figure 2.2: S-PLUS Workbench **History** view

History view control and right-click menus

You can use the **History** view control menu (click ) to:

- Select input displayed in the **History** view and copy it to the **Console**.
- Clear the **History** view.

Note

In the **Preferences** dialog, you can set the option to persist entries in the **History** view between sessions. For more information about this option, see the section Store Console History Between Sessions on page 16. The **History** view holds up to 150,000 lines of commands.

The drop-down control menu and the right-click context-sensitive menu are identical in the **History** view.

Objects view

The **Objects** view is similar to the Object Explorer in the S-PLUS GUI. It displays all objects for projects associated with the workspace in two panes: a table view and an expandable tree view (the Workbench Object Explorer).

The two panes of the **Objects** view are linked: when the **Objects** view table pane has focus, items you select in the table are highlighted in the tree pane. When the tree pane has focus, objects you select in the tree are also highlighted in the table pane. (If you select an object member in the tree pane, its corresponding object is highlighted in the table pane.)

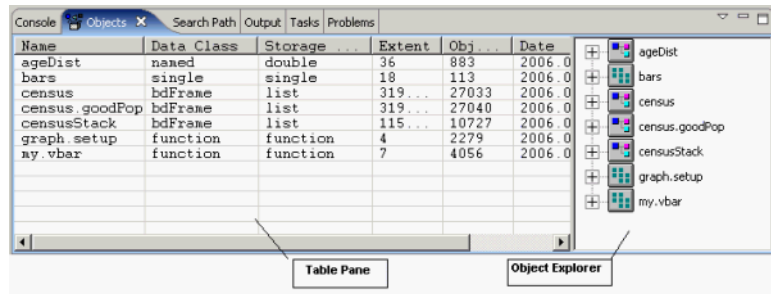


Figure 2.3: S-PLUS Workbench **Objects** view.

Objects view control and right-click menus


You can use the **Objects** view control menu (click ) to perform the following tasks:

- Select another database.
- Refresh the view on the currently-active database.

- Remove the selected object from the currently-active database.
- Show or hide S-PLUS system objects, such as `.Last.value`, `.Data`, and `.Random.seed`. (These objects are hidden by default.)
- Change the number of items displayed in the tree view members.

The drop-down control menu and the right-click context-sensitive menu are identical in the **Objects** view.

Note

When you run code that creates objects in an S-PLUS script, the **Objects** view is not automatically refreshed to display the new objects. To refresh **Objects** view and display newly-created objects, right-click the **Objects** view (or click the control menu button ) , and then from the menu, click **Refresh**.

Warning

If you select a large database, such as `splus`, in the **Objects** view, it can take a long time to display the contents in the table and tree view panes.

Objects view table pane

The **Objects** view includes a table pane displaying a list of the names and types of objects in S-PLUS databases. The **Objects** view table includes the following information about each object:

- name
- data class
- storage mode
- extent
- size
- creation or change date.

By default, the S-PLUS system objects, such as `.Random.seed` and `.Last.value` are hidden. You can display these objects by toggling the option on the **Objects** view control menu.

**Object Explorer
(tree view pane)**

The **Objects** view includes an expandable tree view, the **Object Explorer**. (See Figure 2.4.) Objects listed in the **Object Explorer** correspond to objects in the **Objects** view table pane.

The **Object Explorer** displays icons representing the type of object or object member, along with its name as a label. (These icons are the same icons used in the standard S-PLUS GUI.)

You can expand the objects to display each objects' members. By default, the **Object Explorer** displays up to 25 object members at each expandable level. You can change this default using the **Objects** view context-sensitive menu item, **Set Max Children**.

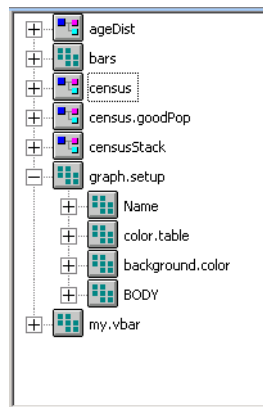


Figure 2.4: *The S-PLUS Workbench Object Explorer.*

- Display the **Set Max Children** dialog to indicate the number of object members to display in the **Object Explorer**. By default, this option is set to 25.

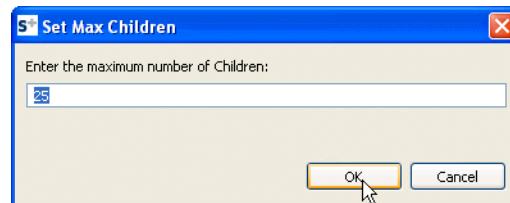


Figure 2.5: *Set Max Children dialog.*

Problems view The **Problems** view is a standard Eclipse view that displays errors as you edit and save code. For example, if you forget a bracket or a quotation mark, and then save your work, the description appears as a syntax error in the **Problems** view.

Note

Syntax problems appear in the **Problems** view only after you save the file.

If your code has a problem that is displayed in the **Problems** view, and the view is not the active view, the **Problems** view tab title appears as bold text.

To open the Script editor at the location of the problem, double-click the error in the **Problems** view.

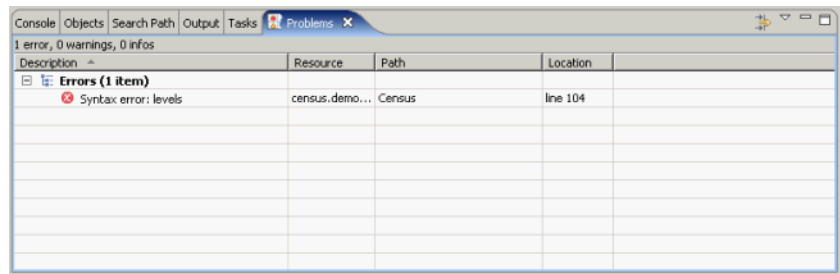



Figure 2.6: S-PLUS Workbench **Problems** view showing the right-click context-sensitive menu.

Problems view control and right-click menus

You can use the **Problems** view control menu (click ) to perform the following tasks:

- Display the **Sorting** dialog to sort the problems displayed in the view, either in ascending or descending order, and according to the problems' characteristics.
- Display the **Filters** dialog to specify properties for filtering problems.

For more information about using these dialogs, see the Eclipse *Workbench User Guide*.

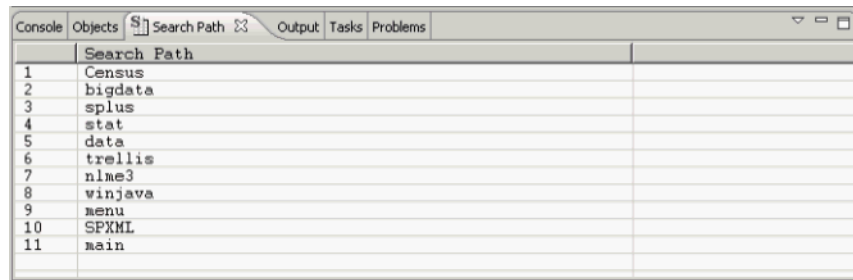
You can use the **Problems** view right-click context-sensitive menu (see Figure 2.6) to perform the following tasks:

- Jump to the location in the file containing the problem.
- Display the file name containing the problem in the **Navigators**. (This action opens an instance of the **Navigators**.)
- Copy the **Problems** view table to the clipboard.
- Select all entries in the table.
- View the properties of the problem.

These menu items are standard to the Eclipse GUI. For more information, see the Eclipse *Workbench User Guide*.

Search Path view

The **Search Path** view displays the names and search path position of all the attached S-PLUS databases.



	Search Path
1	Census
2	bigdata
3	splus
4	stat
5	data
6	trellis
7	nlme3
8	winjava
9	menu
10	SPXML
11	main

Figure 2.7: S-PLUS Workbench **Search Path** view.

The databases that are in your search path determine the objects that are displayed in **Objects** view. That is, if a database is in your search path, the objects in that database appear in the **Objects** view. See the section Examining Objects on page 132. For more information about working with the **Search Path** view, see the section Changing Attached Databases on page 125.

The first position in the **Search Path** view shows the current working directory, which can be either the workspace or the current path. You can set a project to be the working project by right-clicking its name in the **Navigators**, and then clicking **Toggle Working S-PLUS Project**. See the section Navigators on page 39, and the section Setting the Working Project on page 123.

Search Path view control and right-click menus

You can use the **Search Path** view control menu (click ) to:

- Attach a library.
- Attach a module.
- Attach a directory.
- Detach the currently-selected database in the view.
- Refresh the current view.

The drop-down control menu and the right-click context-sensitive menu are identical in the **Search Path** view.

Note

When you use the control menu to add to (or remove from) the **Search Path** view a library, module, or directory, the view automatically refreshes. When you run code to add or remove a library, module, or directory, the view is not automatically refreshed. To refresh the view, right-click the **Search Path** view (or click the control menu button, and then from the menu, click **Refresh**).

Tasks view

The **Tasks** view is a standard Eclipse IDE view, which is customized in S-PLUS to provide three levels of tasks:

Table 2.2: *S-PLUS Workbench Tasks.*

Task	Description
FIXME	Defines high-priority tasks. The task appears with an exclamation mark in the Tasks view.
TODO	Defines medium-priority tasks.
XXX	Defines low-priority tasks.

You can change these tasks, or you can add your own custom tasks. For more information about changing task settings, see section Task Options on page 22, and the section To set the S-PLUS preferences on page 111.

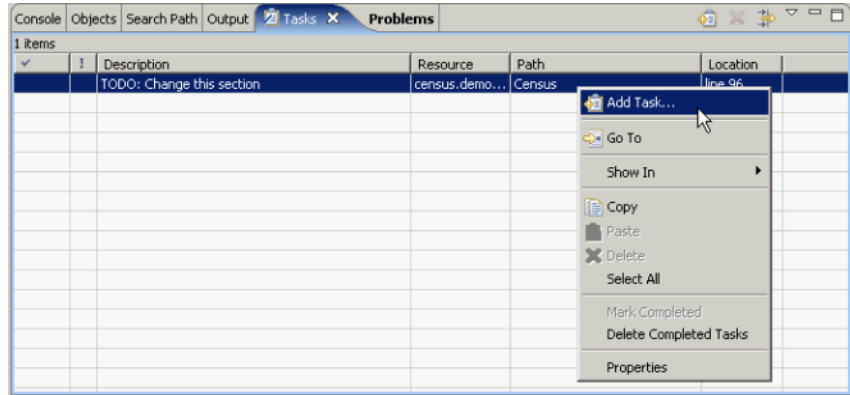





Figure 2.8: S-PLUS Workbench **Tasks** view showing the right-click context-sensitive menu.


Tasks view toolbar

The **Tasks** view also contains a toolbar that displays the following buttons:

Table 2.3: **Tasks** view buttons.

Button	Description
	Click to display the Add Task dialog to add a custom task.
	Click to delete the selected custom task. (Note that you cannot use this button to delete tasks identified in the script.)
	Click to display the Filters dialog to specify properties for filtering the tasks.

Tasks view control and right-click menus

You can use the **Tasks** view control menu (click ) to perform the following tasks:

- Display the **Sorting** dialog to sort the tasks displayed in the view, either in ascending or descending order, and according to the tasks' characteristics.
- Display the **Filters** dialog to specify properties for filtering tasks.

You can use the standard Eclipse **Tasks** view right click context-sensitive menu to:

- Add a task to the list that is not linked to a file (Displays the Eclipse **Add Task** dialog).
- Open a file and display the location of a linked task in the Script Editor.
- Display the location of a linked task in the **Navigator** (opens an instance of the **Navigator**).
- Copy the **Tasks** view table to the clipboard.
- Select all entries in the table.
- Delete all tasks marked as completed (that is, containing a check mark in the first column).
- View the properties of the task.

For more information about the basic Eclipse **Tasks** view, see the Eclipse *Workbench User Guide*.

S-PLUS WORKBENCH DEBUG PERSPECTIVE

3

Introduction	66
Debug Perspective Options and Preferences	68
Setting Preferences	69
Debug Mode	69
Debug Run Menu Options	70
Debug Perspective Views	74
Profiler	96
Profiler views	97

INTRODUCTION

The S-PLUS Workbench includes the Debug perspective, which is based on the Eclipse standard debugging perspective.

From the Debug perspective, you can observe the run-time behavior of your program and determine the location of semantic errors. The S-PLUS debugger understands features that are built into the S-PLUS programming language and its associated libraries. With the S-PLUS debugger, you can break (suspend) execution of your program to examine your code and evaluate variables.

After you have written your code and resolved any syntax errors, you can use the S-PLUS debugger to correct any logic errors that keep your code from running correctly. Using the S-PLUS debugger, you can:

- Control your code testing by setting break points, stepping into, through, and out of code, and pausing or terminating the process at any point using the S-PLUS debugger features.
- Set, disable, enable, or remove breakpoints while you are debugging.
- View variable and expression values at breakpoints while stepping through your code.
- Track resource allocation and function use.

Figure 3.1 shows the Debug perspective's views. This chapter describes the options, features, and views included in the Debug perspective.

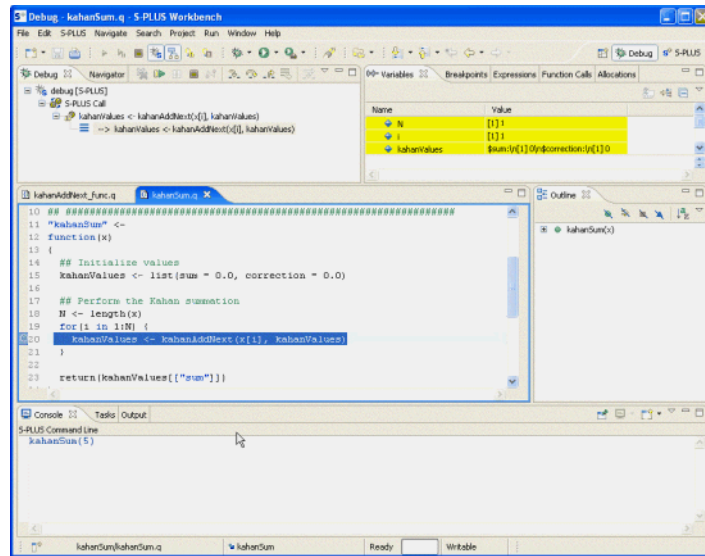


Figure 3.1: *The Debug perspective.*

The Debug perspective also includes a profiler, which you can use to inspect allocated memory and functions called, including call count and duration. For more information about the S-PLUS Profiler, see the section Profiler on page 96.

- For tasks that walk you through using the S-PLUS debugger and profiler, see the section Chapter 4, S-PLUS Workbench Tasks.
- For information about the S-PLUS perspective, see Chapter 2, The S-PLUS Perspective.

Note

You can create your own perspective that displays a combination of views from the perspectives, or you can change the Debug perspective to suit your development style by adding, moving, hiding, or closing views. For more information about customizing the views within the perspective, see the section Changing the S-PLUS Workbench Perspective on page 31, or see the section Customized Perspective Views on page 120.

DEBUG PERSPECTIVE OPTIONS AND PREFERENCES

When you examine the Debug perspective, examine the S-PLUS Workbench toolbars, menus, default options, and preferences in the IDE.

Note that the S-PLUS Workbench toolbar includes the S-PLUS debugger buttons (as well as the **Profiler** button). These buttons are described in greater detail in Table 1.4.

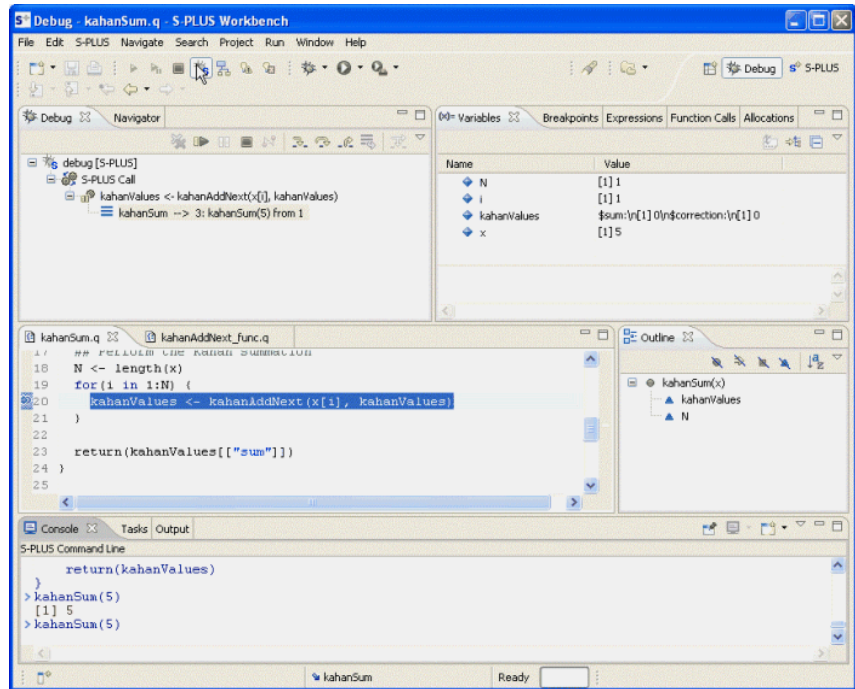


Figure 3.2: The S-PLUS Workbench toolbar.

Note

When you are in the Debug perspective, notice that the Eclipse environment displays a generic toolbar that includes a **Run** button, a **Debug** button, and an **External Tools** button. These buttons might work with other Eclipse plug-ins, but they are not intended to be used with S-PLUS. You can set breakpoints from the **Debug** view toolbar, or from several menus, and you can run code using the **Run S-PLUS Code** button on the S-PLUS toolbar or from the console.


Setting Preferences


From the menu, click **Window ► Preferences** to open the **Preferences** dialog and examine the options. (For more information about setting preferences, see the section *Examining S-PLUS Preferences* on page 12. For more information about Eclipse preferences, see the Eclipse *Workbench User's Guide*, available from the **Help ► Help Contents** menu in the IDE.)

Most options in the **S-PLUS** pages of the **Preferences** dialog apply to global settings in the S-PLUS Workbench. For example, options controlling editor or **Console** view text colors apply to both perspectives. Only the **Profiler** page under **S-PLUS** controls S-PLUS debugger behavior, and that controls only the refresh rate for system allocations and function calls. See section *Examining S-PLUS Preferences* on page 12 for more information.


Debug Mode

To start debugging, first activate the debugger using one of the following methods:

- On the toolbar, click **Toggle S-PLUS Debugger** .
- On the menu, click the **Run ► Toggle S-PLUS Debugger**.
- On the keyboard, press CTRL+ALT+D.

After you activate the S-PLUS debugger, any expression you type in the **Console** view, or that you run by clicking **Run S-PLUS Code**  on the toolbar, invokes the S-PLUS debugger.

Note

You can set Eclipse an option to be notified that a debug session is about to begin (that is, if you click **Debug** ) and try to run a function in the **Console** view that encounters any breakpoints).

1. From the main menu, click **Windows ► Preferences**.
2. Expand **Run/Debug** and select **Perspectives**.
3. In the **Perspectives** dialog, in the **Open the associated perspective when launching** group, select **Prompt**. Click **OK**.

Using this Eclipse option, you are prompted to change to the Debug perspective with the message box shown in Figure 3.3. Clicking **Yes** displays the Debug perspective with the **Debug** view open and the debugging started.

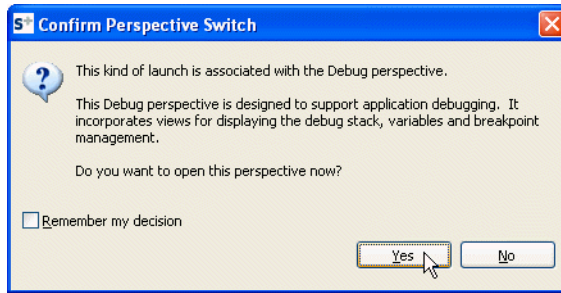


Figure 3.3: The *Confirm Perspective Switch* message box.

Debug Run Menu Options

When you switch to the Debug perspective, the S-PLUS Workbench **Run** menu changes to list all of the code control actions specific to that perspective. Note that many of the options listed in this menu are default Eclipse debugging options. For more information about those options, see the *Eclipse Workbench User Guide*. The Debugger actions are available in the Debug perspective views.

Table 3.1: *Debug perspective Run menu.*

Menu item	Description
Run S-PLUS Code	Runs the code in the currently-active file, or runs the selected code.
Run Next S-PLUS Command	Runs the next available S-PLUS command.
Toggle S-PLUS Debugger	When toggled on, engages the S-PLUS debugger. (You can engage the Debugger in either the S-PLUS or the Debug perspectives; however, by default, the views displaying debugging information are visible in the Debug perspective.)

Table 3.1: *Debug perspective **Run** menu. (Continued)*

Menu item	Description
Toggle S-PLUS Profiler	<p>When toggled on, engages the S-PLUS Profiler. (You can engage the S-PLUS Profiler in either the S-PLUS or the Debug perspectives; however, by default, the views displaying profiling information are visible in the Debug perspective.)</p> <p>You do not need to engage the debugger in order to engage the Profiler. See the section Profiler on page 96 for more information.</p>
Resume	Resumes debugging when the debugger is paused.
Suspend	Suspends debugging.
Terminate	Terminates debugging.
Step Into	Steps into the current function by one level
Step Over	Stays at the same expression level but steps to the next expression.
Step Return	Steps out of the current function by one level.
Run to Line	Core Eclipse debugger option; not implemented in S-PLUS.
Use Step Filters	Core Eclipse debugger option; not implemented in S-PLUS.
Run Last Launched	Core Eclipse debugger option; not implemented in S-PLUS.

Table 3.1: *Debug perspective **Run** menu. (Continued)*

Menu item	Description
Debug Last Launched	Core Eclipse debugger option; not implemented in S-PLUS.
Run Last Launched	Core Eclipse debugger option; not implemented in S-PLUS.
Debug Last Launched	Core Eclipse debugger option; not implemented in S-PLUS.
Run History	Core Eclipse debugger option; not implemented in S-PLUS.
Run As	Core Eclipse debugger option; not implemented in S-PLUS.
Run	Core Eclipse debugger option; not implemented in S-PLUS. (Use the Run S-PLUS Code option at the top of the main menu, F9, the Debug view context-sensitive menu, or on the Debugger toolbar.)
Debug History	In its submenu, lists the previously-launched debugging actions. From this list, you can select a previous
Debug As	Core Eclipse debugger option; not implemented in S-PLUS.
Debug	Core Eclipse debugger option; not implemented in S-PLUS.
External Tools	Core Eclipse debugger option; not implemented in S-PLUS.

Table 3.1: *Debug perspective **Run** menu. (Continued)*

Menu item	Description
Toggle S-PLUS Warning Breakpoint	Requires that the S-PLUS debugger be toggled on. When toggled on, stops execution if S-PLUS encounters a warning. See Table 3.7 in the section Breakpoints view on page 89 for more information about warning breakpoints.
Toggle S-PLUS Error Breakpoint	Requires that the S-PLUS debugger be toggled on. When toggled on, stops execution if S-PLUS encounters an error. See Table 3.7 in the section Breakpoints view on page 89 for more information about error breakpoints.
Toggle Line Breakpoint	When toggled on, removes the breakpoint on the selected line.
Toggle Method Breakpoint	Core Eclipse debugger option; not implemented in S-PLUS.
Toggle Watchpoint	Not implemented in the debugger.
Skip All Breakpoints	When selected, disregards but maintain (that is, does not remove or disable) all breakpoints. When this button is toggled on, all breakpoints appear with a diagonal slash, as shown in the button.
Remove All Breakpoints	Removes every breakpoint from files in open projects. (This item does not remove breakpoints from files in closed projects.)

DEBUG PERSPECTIVE VIEWS

The Debug perspective includes views specific to using the debugger and the profiler, as well as views shared across perspectives. For a list of all views and their default perspectives, see Table 1.6 in Chapter 1. (Chapter 1 also includes descriptions of the shared views.)

The Debug perspective includes the default Eclipse **Navigator** view and customized views. Customized views in the Debug perspective include the following:

Table 3.2: *Debug perspective views and exercise references.*

View	Descriptions and Practice exercises
Allocations view	An S-PLUS Profiler view. For a description, see the section Allocations view on page 98. For practice using this view, see the exercise in the section Examining Resource Usage on page 150. (The Profiler views are discussed in more detail in the section Profiler Mode on page 97.)
Breakpoints view	For a description, see the section Breakpoints view on page 89. For practice using this view, see the exercise in the section Setting breakpoints on page 141.
Console view	Shared view. For a description, see the section S-PLUS Workbench Console on page 37. For practice using this view, see the exercise in the section To run copied script code on page 136.
Debug view	For a description, see the section Debug view on page 76. For practice using this view, see the exercise in the section Examining the call stack on page 144.
Expressions view	For a description, see the section Expressions view on page 87. For practice using this view, see the exercise in the section Examining Variables and Expressions on page 145

Table 3.2: *Debug perspective views and exercise references. (Continued)*

View	Descriptions and Practice exercises
Function Calls view	An S-PLUS Profiler view. For a description, see the section Function Calls view on page 97. For practice using this view, see the exercise in the section Examining Function Calls on page 150. (The Profiler views are discussed in more detail in the section Profiler Mode on page 97.)
Outline view	Shared view. For a description, see the section Outline view on page 41. For practice using this view, see the exercise in the section To examine the outline on page 131.
Output view	Shared view. For a description, see the section Output on page 43. For practice using this view, see the exercise in the section To run code on page 137.
Variables view	For a description, see the section Variables view on page 82. For practice using this view, see the exercise in the section Examining Variables and Expressions on page 145.
Tasks view	Shared view. For a description, see the section Tasks view on page 61. For practice using this view, see the exercise in the section Adding a Task to A Script on page 132.

Additionally, the Debug perspective displays the Script Editor, which is shared with the S-PLUS perspective. See the section Editor on page 79 for more information about using the Script Editor with the Debugger. See the section S-PLUS Workbench Script Editor on page 45 for more general information about editing code in the Script Editor.

From the Debug perspective, you can observe the run-time behavior of your program and determine the location of semantic errors. The Workbench debugger understands features that are built into the

S-PLUS programming language and its associated libraries. With the debugger, you can break (suspend) execution of your program to examine your code and evaluate and edit variables.

Debug view

The **Debug** view displays the call stack of a currently-paused expression. Clicking any level of the call stack displays in the **Editor** the current function and/or the highlighted expression.

Figure 3.4 displays the **Debug** view, in its default position, displaying the call stack for the `kahanSum` example.

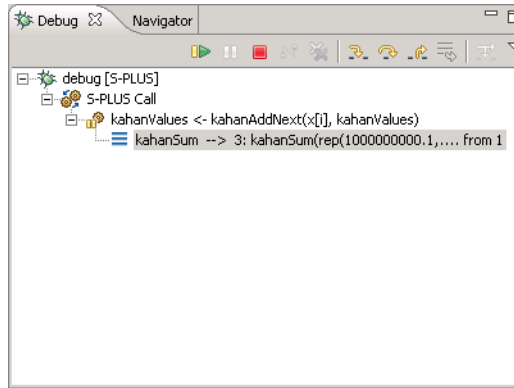


Figure 3.4: *The Debug view.*

Debug view toolbar

The **Debug** view contains a toolbar with the following buttons for evaluation control, in the order of their appearance, left to right:

Table 3.3: *Debug view toolbar buttons.*










Button	Description
	Remove All Terminated Launches. Clears the call stack of all debugging sessions that ended with a termination.
	Resume. Continues to the next breakpoint.
	Suspend. Pauses the execution as though a breakpoint had been hit.
	Terminate. Stops the execution. Similar to ESC functionality.
	Disconnect. For remote debugging. Not implemented for S-PLUS.
	Step Into. Steps into the current function by one level.


Table 3.3: Debug view toolbar buttons. (Continued)

Button	Description
	Step Over. Stays at the same expression level but steps to the next expression.
	Step Return. Step out of the current function level.
	Use Step Filters/Step Debug. This feature is not supported in the S-PLUS Workbench.

Note

The feature **Drop to Frame** is not implemented in the S-PLUS Workbench.

Debug view control and right-click menus

The **Debug** view contains a control drop-down () menu with one command: **View Management**, which displays the **View Management** page of the Eclipse **Preferences** dialog, in which you can set options to open and close views automatically. This dialog is also available from the **Windows ► Preferences** menu. For more information about using this menu item, see the Eclipse *Workbench User's Guide*.

You can use the **Debug** view right-click menu (Figure 3.5) to perform the following tasks:

- Copy the contents of the stack.
- Step into the code.
- Step over the code.
- Step one level out of the current function.
- Resume debugging.
- Suspend debugging.

- Terminate the debugging session.
- Terminate and restart the current debugging session.
- Remove from the view all previously terminated debugging sessions.
- Terminate and remove the currently-active debugging session.
- Restart the current debugging session.
- Terminate all debugging.

These menu items are available on the toolbar, or from the main **Run** menu. For more information, see the section Debug view toolbar on page 77 or the section Debug Run Menu Options on page 70.

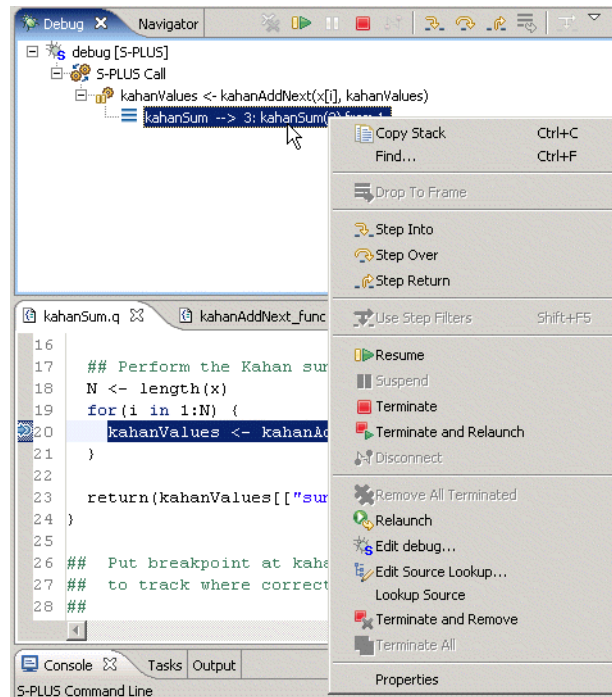


Figure 3.5: The *Debug* view right-click context-sensitive menu.

Editor

The Debugger perspective uses the existing S-PLUS Workbench editor. You can set and remove breakpoints in the Script Editor by:

- Double-clicking the margin on the left side of the screen (to the left of the line numbers).

- By right clicking the margin, and from the menu, select **Toggle Breakpoint**.
- By using the **Run ► Toggle Line Breakpoint** menu option.
- By pressing CTRL+SHIFT+B.

When you are debugging, if your functions call any functions in files other than those in your workspace (including functions in a library), you can double-click the expression in the **Debug** view and open a temporary file that contains the called function. You can set breakpoints in these functions, too.

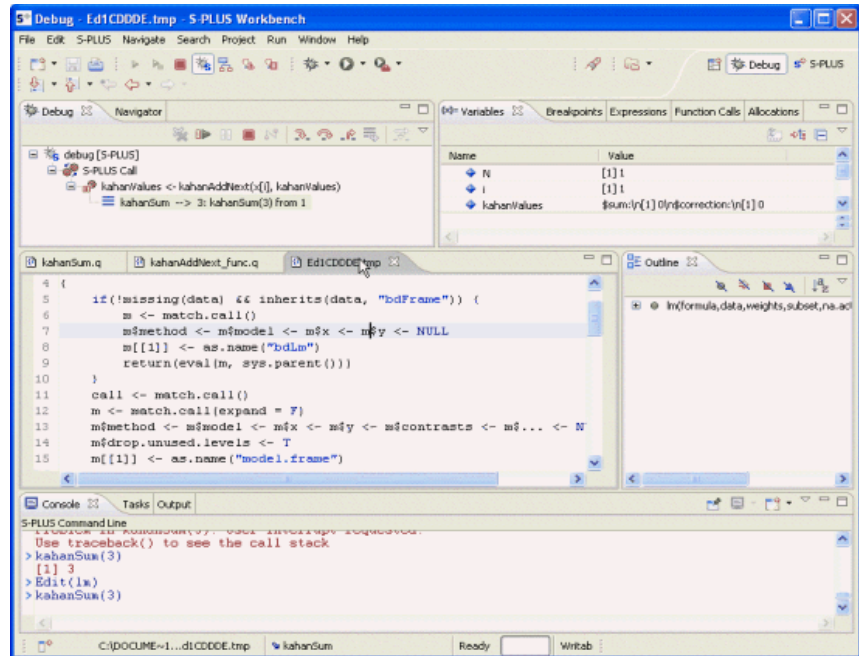


Figure 1: A temporary file in the debugger.

You can view functions that are not defined in your workspace in one of the following ways:

- Double-click the **Debug** view.
- Press CTRL+click in the S-PLUS Script Editor.
- On the menu, click **S-PLUS ► Find**

- Press CTRL+SHIFT+F

Note

- Breakpoints that you set in functions in your workspace are associated with function *and* with the file. These breakpoints persist until you remove them.
- Breakpoints that you set in functions outside of your workspace are associated with the functions, and not with the temporary files. They persist until you remove them.
- Setting breakpoints in code files in the S-PLUS Workbench does not affect the file if you open it in the S-PLUS GUI in Windows. Breakpoints are evaluated only in the S-PLUS Workbench, and only when the debugger is engaged.
- Breakpoints can be set only on a line contained within a function definition. Lines not contained within a function cannot have a breakpoint set.

If you close a temporary file containing a breakpoint, and then rerun your function, the functions called by your code reopen in another temporary file, and any breakpoints you set persist.

Examining Expression Values in Tooltips

Using the **Hover** feature, you can position the mouse over an expression in the Script Editor, and then examine the expression's value, which appears in a tooltip. This feature is available for all expressions in the Script Editor, not just those where a breakpoint appears; however, examining the value of an expression at a breakpoint can be very useful.

You can limit the size of the expression that the **Hover** feature evaluates by using the following S-PLUS command:

```
options(workbenchMaxDims=c(rows, columns))
```

See the section **Hover** on page 18 for more information.

You can enable or disable the hover tooltip feature in the **Editor** options dialog from the **Windows ► Preferences** menu. This feature is enabled by default.

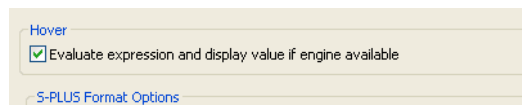


Figure 3.6: *Hover option in the Editor preferences dialog.*

For practice tasks on setting breakpoints, see the section Setting breakpoints on page 141.

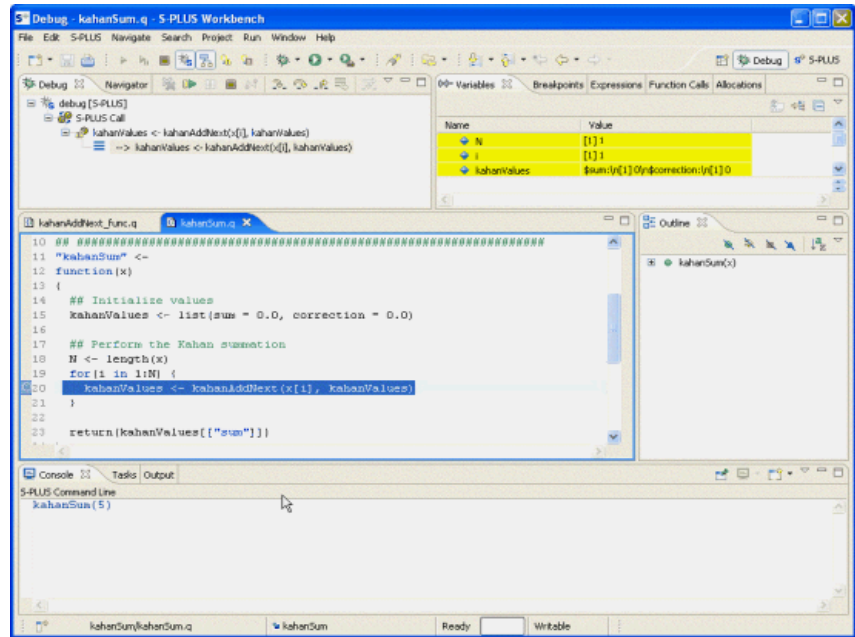


Figure 3.7: *Breakpoints view and Editor.*

Variables view

Displays all variables in the current frame. As you debug, at each breakpoint or step, the debugger re-evaluates the variables. At any breakpoint or stopping point, you can review, but not edit or alter, the variables at the current frame.

Figure 3.8 shows the **Variables** view with the current variable selected. The **Details** pane of this view contains variable information that would result from calling `print()` on the selected variable or

expression. The **Details** pane is editable; you can select, cut, or copy the contents of this pane. Editing the **Details** pane does not affect the value of a variable.

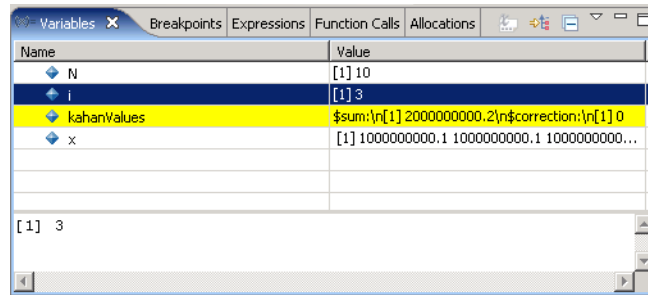







Figure 3.8: The *Variables* view.

Variables view and Expressions view toolbars

The **Variables** view and Expressions view contain similar toolbars to control the view display and feature options.

Table 3.4: *Variables* view and *Expressions* view toolbar buttons.

Button	Description
	Show Type Names. Select to display the variables' types.
	Show Logical Structure. This feature is currently not supported in the S-PLUS Workbench.
	Collapse All. Collapses the logical structure display (which is currently not supported in the S-PLUS Workbench).
	Remove Selected Expressions (Expressions view only).
	Remove All Expressions (Expressions view only).

Variables view control and right-click menus

The **Variables** view and **Expressions** view drop-down control menus provide additional options to control the view's display. The respective menus are available from the down arrow button on the **Variables** or **Expressions** view toolbar.

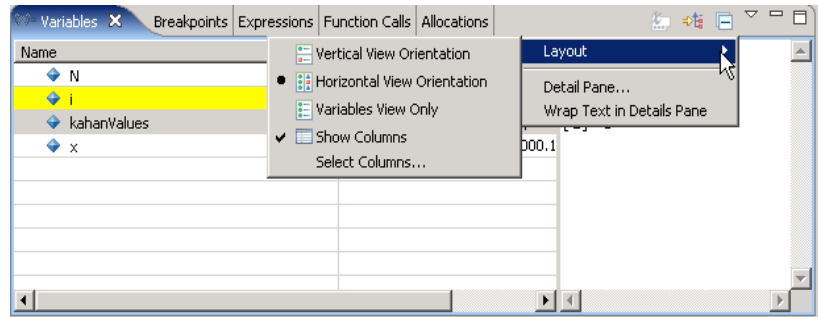


Figure 3.9: The *Variables* view control menu.

The **Variables** view and **Expressions** view control menus include the following options:

Table 3.5: *Variables* view and *Expressions* view control menu options.

Menu item	Description
Vertical View Orientation	Tiles the Details pane of the view vertically. That is, the Details pane appears below the Variables or Expressions pane.
Horizontal View Orientation	Tiles the Details pane of the view horizontally. That is, the Details pane appears beside the Variables or Expressions pane.

Table 3.5: *Variables* view and *Expressions* view control menu options.

Menu item	Description
Variables View Only Expressions View Only	Hides the Details pane of the Variables or Expressions view.
Detail Pane	Displays the Configure Details Area dialog, which controls the maximum number of characters to display in the Details pane. See Figure 3.10.
Wrap Text in Details pane	Wraps the text that appears in the Details pane.

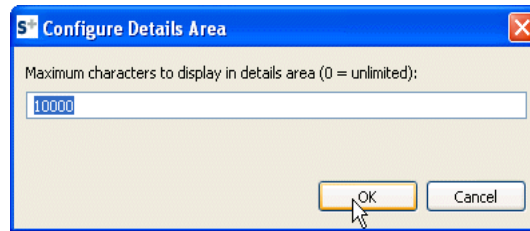
**Figure 3.10:** *Configure Details Area* dialog.**Note**

Figure 3.10 shows the **Configure Details Area** dialog, with which you can set the number of characters to display. This option just controls the number *displayed*; it does not limit the number of characters *returned*. To limit the number of text variables and expressions to return, use the S-PLUS command `options(workbenchMaxDims=c(rows, columns))`. This option is useful if you are working with a large number of text variables or expressions.

Setting this option also limits the size of the expression that the hover feature evaluates. For example, if you are evaluating a large data object, and you hover the mouse over the expression, if you do not set this option, S-PLUS tries to evaluate the expression on the spot.

The **Variables** view contains two right-click context-sensitive menus:

- The **Variables** view (Figure 3.9).

- The **Details** pane (Figure 3.12).

You can use the **Variables** view right-click context-sensitive menu to perform the following tasks:

- Select all variables in the pane.
- Copy the selected variable.
- Find a specified variable.
- Set an expression watch for the selected variable. (When you select this option, the selected variable is added to the **Expressions** view.)

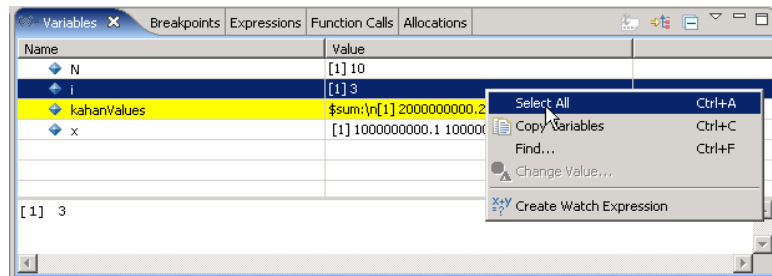


Figure 3.11: *Variables* view showing the right-click menu.

You can use the right-click context-sensitive menu in the **Variables** view **Details** pane to perform the following tasks:

- Cut the currently-selected text.
- Copy the currently-selected text.
- Paste the contents of the clipboard to the cursor location in the pane.
- Select all text in the pane.

- Find a specified string in the pane. (The S-PLUS Workbench does not support replacing strings in the **Details** pane using the **Find/Replace** dialog.)

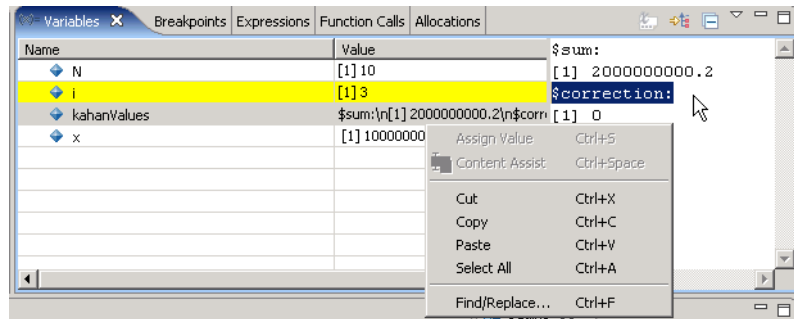


Figure 3.12: *Variables* view showing the right-click menu in the *Details* pane.

Expressions view The **Expressions** view displays the values of any S-PLUS expression. Like the **Variables** view, it is re-evaluated at each evaluation pause (breakpoint or step).

Note on Expressions

An *expression* is any syntactical interaction that S-PLUS can evaluate. Expressions persist from session to session. S-PLUS recognizes a wide variety of expressions, but in interactive use, the most common are names, which return the current definition of the named data object, and function calls, which carry out a specified computation. Any of the following are S-PLUS expressions:

```
1:10
rnorm(5)
mean(1:10)
traceback()
```

If you were debugging a function, for example:

```
incrementByTwo <- function(x) {
  * x + 2
}
```

you could have an expression that evaluated:

```
x + 2
```

at the breakpoint (denoted with the * in the above function definition).

Note

If you leave in the **Expressions** view expressions that are no longer in scope for your current debugging session, you might notice that the debugger slows significantly to evaluate the expression that is no longer in scope. To keep the debugger from slowing down, remove expressions that are no longer in scope for your current debugging session.

For more information about expressions, see the *Programmer's Guide*, or see the S-PLUS Help topic **ExpressionLanguage**.

The **Expressions** view toolbar buttons are the same as those of the **Variables** view, with the addition of the **Remove** and **Remove All** buttons. See Table 3.4 for more information.

The **Expressions** view contains two right-click context-sensitive menus:

- The **Expressions** view (Figure 3.13)
- The **Details** pane (Figure 3.12).

You can use the **Expressions** view right-click context-sensitive menu to perform the following tasks:

- Select all expressions in the pane.
- Copy the selected expression.
- Remove the selected expression.
- Remove all expressions in the view.
- Add an expression to watch (opens the **Add Watch Expression** dialog, in which you can provide an expression and indicate whether to enable or disable it).
- Re-evaluate the expressions.
- Disable the currently-selected expression.
- Enable the currently-selected expression (if it was previously disabled).

- Edit the currently-selected expression. (Opens the **Edit Watch Expression** dialog, in which you can change expression and indicate whether to enable or disable it.)

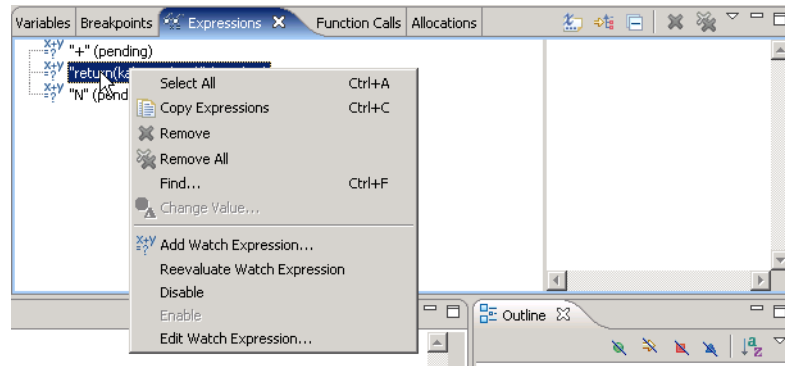


Figure 3.13: *Expressions* view showing the right-click menu.

The right-click context-sensitive menu for the **Details** pane in the **Expressions** view is the same as that of the **Variables** view **Details** pane. See Figure 3.12 and the section Variables view control and right-click menus on page 84 for more information.

Find a specified string in the pane. (The S-PLUS Workbench does not support replacing strings in the **Details** pane using the **Find/Replace** dialog.)

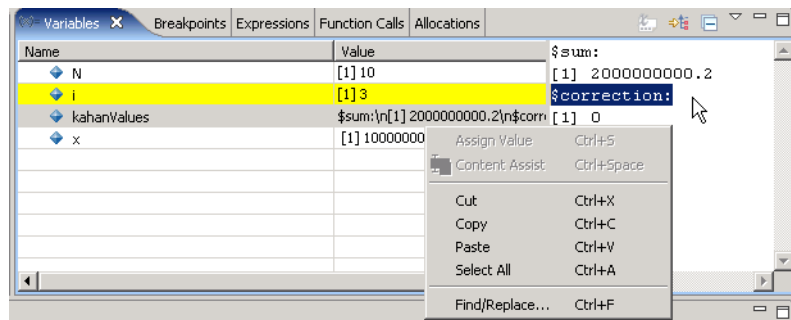


Figure 3.14: *Variables* view showing the right-click menu in the **Details** pane.

Breakpoints view The **Breakpoints** view displays the currently set breakpoints, which you can organize by resources, files, working sets, or just a simple list or type. Each breakpoint displayed in the **Breakpoints** view shows

the function name (e.g., `kahanAddNext`), the file name (e.g., **`kahanAddNext_func.q`**), and the line number (e.g., `[line 7]`) where the breakpoint occurs.

In addition to setting general user interface options, you can use the **Breakpoints** view to manage breakpoint working sets and group breakpoints. See the Eclipse *Workbench User's Guide* for more information.

Selecting a breakpoint displays in the **Editor** the associated file, highlighting the breakpoint line. You can activate, disable, or delete breakpoints from this view.

Figure 3.15 displays the **Breakpoints** view with the file structure shown, and all breakpoints activated.

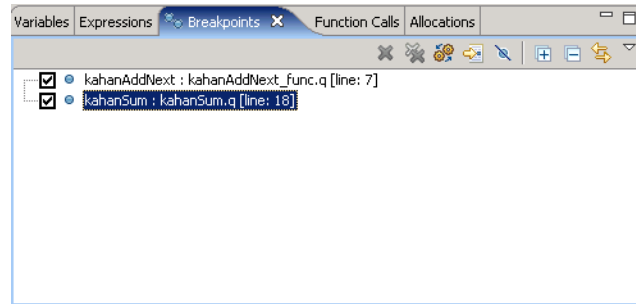


Figure 3.15: *Breakpoints* view.

Breakpoint types

Breakpoints are the best tools to stop an evaluation and inspect the engine's state. The S-PLUS Workbench supports three types of breakpoints.

Table 3.6: *Types of breakpoints.*









Breakpoint type	Description
Line breakpoints	<p>Use line breakpoints to stop an evaluation at the specified line number. To set line breakpoints, from any perspective:</p> <p>Double-click the left margin of the S-PLUS Editor.</p> <p>Right-click the left margin of the S-PLUS Editor.</p> <p>From the Debug perspective, Click the Run ► Toggle Line Breakpoint menu item.</p> <p>After you set a line breakpoint, you can enable or disable it in the Breakpoints view, or by right-clicking the breakpoint marker () in the left margin of the Editor.</p> <p>(For more information about using the Breakpoints view, see the section Breakpoints view on page 89.)</p>
Warning breakpoints	<p>Warning breakpoints are triggered only if the S-PLUS Debugger is toggled on.</p> <p>Use warning breakpoints to stop an evaluation when a warning is generated. You can activate warning breakpoints from any perspective by clicking Toggle S-PLUS Warning Breakpoint () on the S-PLUS toolbar, or by clicking the Run ► Toggle S-PLUS Warning Breakpoint menu item.</p> <ul style="list-style-type: none"> Warning breakpoints do not appear in the Breakpoints view. Warning breakpoints are not affected by the option Skip All Breakpoints ().






Table 3.6: *Types of breakpoints. (Continued)*

Breakpoint type	Description
Error breakpoints	<p>Error breakpoints are triggered only if the S-PLUS Debugger is toggled on.</p> <p>Use error breakpoints to stop an evaluation when an error is generated. You can activate error breakpoints from any perspective by clicking Toggle S-PLUS Error Breakpoint () on the S-PLUS toolbar, or by clicking the Run ► Toggle S-PLUS Error Breakpoint menu item.</p> <ul style="list-style-type: none"> Error breakpoints do not appear in the Breakpoints view. Error breakpoints are not affected by the option Skip All Breakpoints ()

Breakpoints view toolbar

The **Breakpoints** view contains a toolbar to control the view's display and feature options.

Button	Description
	Remove Selected Breakpoints. From the Breakpoints view, click to remove the selected breakpoint from both the Debug view and the Breakpoints view.
	Remove All Breakpoints. From the Breakpoints view, click to remove every breakpoint from both the Debug view and the Breakpoints view.
	Show Breakpoints Supported by Selected Target. When toggled off, all breakpoints are displayed. When toggled on, the Breakpoints view displays only breakpoints applicable to the selected debug target. For example, if you had installed a Java package for Eclipse (not included in the S-PLUS Workbench), and you were running a Java debug session and an S-PLUS debug session simultaneously, you could filter using this feature.

Button	Description
	Go to File for Breakpoint. Click to jump to the file and line number containing the breakpoint currently selected in the Breakpoints view.
	Skip All Breakpoints. Click to disregard but maintain (that is, not remove or disable) all breakpoints. When this button is toggled on, all breakpoints appear with a diagonal slash, as shown in the button.
	Expand All. If the Breakpoints view is set to display breakpoints in groups such as files, working sets, projects, resources, or breakpoint types, clicking this button expands the tree to display the breakpoints in all groups. (See Table 3.7 for more information about the group display options.)
	Collapse All. If the Breakpoints view is set to display breakpoints in groups such as files, working sets, projects, resources, or breakpoint types, clicking this button collapses the tree to display only the top-level groups. (See Table 3.7 for more information about the group display options.)
	Link With Debug View. As breakpoints are encountered, they are selected in the Breakpoints view.

Breakpoints view control and right-click menus

The **Breakpoints** view contains a control menu to control the types and levels of resources displayed, and options for managing working sets. See the *Eclipse Workbench User's Guide* for more information about managing working sets.

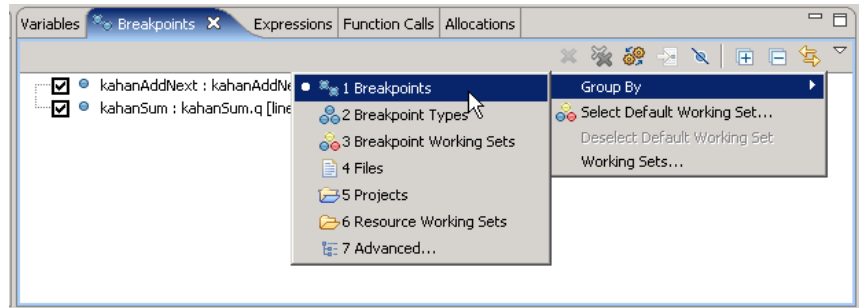


Figure 3.16: *The Breakpoints view menu.*

The **Breakpoints** control menu includes the following options:

Table 3.7: *Breakpoints* view menu.

Menu Item	Description
Group By	<p>Displays a submenu providing the following options:</p> <ul style="list-style-type: none"> • Breakpoints. Displays only the breakpoints in a flat list. • Breakpoint Types. Displays breakpoints grouped by type (Java, S-PLUS, and so on). • Breakpoint Working Sets. Displays breakpoints grouped by identified working sets. See (working sets section) for more information. • Files. Displays breakpoints grouped by the files containing them. • Projects. Displays breakpoints grouped by the projects containing them. • Resource Working Sets. Displays breakpoints by the resources to which they belong. • Advanced. Displays the Group Breakpoints dialog. See the Eclipse <i>Workbench User's Guide</i> for more information about using working sets and groups.
Select Default Working Set	<p>Displays a dialog to create, select, or remove the breakpoint working set that is your project's default. See the Eclipse <i>Workbench User's Guide</i> for more information about using working sets.</p>
Deselect Default Working Set	<p>Clears the working set that you specified in the Select Default Working Set dialog.</p>
Working Sets	<p>Displays the Select Working Set dialog.</p>

You can use the **Breakpoints** right-click context-sensitive menu (see Figure 3.17) for the following tasks:

- Open the file and location for the selected breakpoint.
- Enable the selected disabled breakpoint.
- Disable selected breakpoint.
- Remove the selected breakpoint.
- Remove all breakpoints in the view.
- Select all breakpoints in the view.
- Copy the breakpoints to the clipboard.

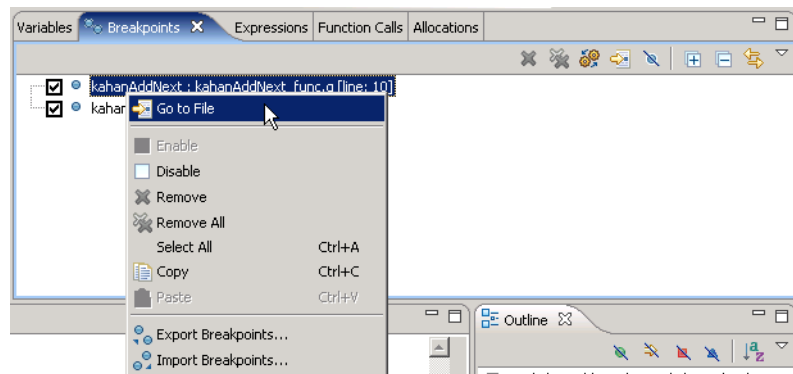


Figure 3.17: The **Breakpoints** view displaying the right-click menu.

Working with Working Sets

The S-PLUS Workbench provides tools to group and manage project files and resources using working sets. Working set menus are available in several Eclipse views, including the **Breakpoints** view. For general information about using working sets, see the Eclipse *Workbench User's Guide*.

Console, Output, and Outline views

The Debugger perspective shares the S-PLUS Workbench **Console** view, **Output** view, and **Outline** view. For more information about using these views, see:

The section S-PLUS Workbench Console on page 37

The section S-PLUS Perspective Views on page 53.

Profiler

The Workbench Profiler is composed of two views: the **Function Calls** view and the **Allocations** view, which are available in the Debug perspective. You can run the Profiler from either the **Run**

menu or from the **Toggle S-PLUS Profiler** button (🔧), located next to the **Toggle S-PLUS Debugger** button on the S-PLUS Workbench toolbar. (See Figure 3.2.)

Profiler Mode

To start profiling, first activate the S-PLUS profiler by clicking **Toggle S-PLUS Profiler** toolbar item, by typing CTRL+ALT+P, or by clicking **Run ► Toggle S-PLUS Profiler** on the menu. Once the Profiler is activated, any expression you type in the **Console** view, or that you enter by clicking **Run S-PLUS Code**, invokes the Profiler adds to the **Function Call** and **Allocation** views.

Profiler views

The S-PLUS Workbench Profiler includes two views to monitor the system performance:

- **Function Calls** view
- **Allocations** view

These views are described in this section.

Function Calls view

By default, the **Function Calls** view displays a function call tree that reflects the engine's activity.

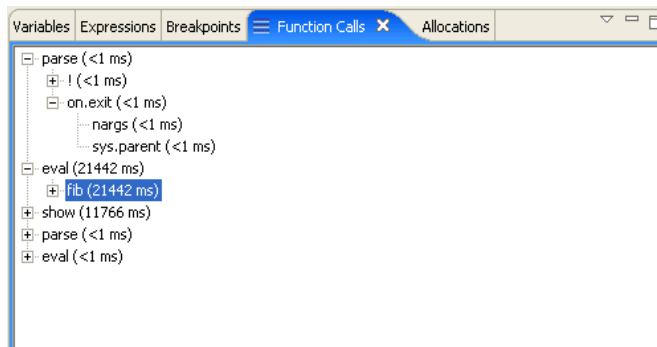
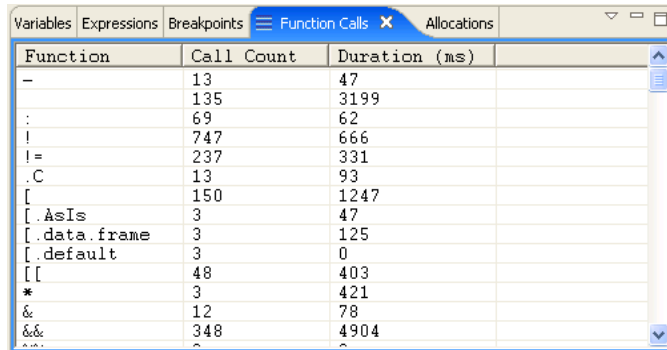


Figure 3.18: *Function Calls* view, tree display.

Alternatively, you can display the information in a tabular view by either of the following methods:

- Right-click the view, and from the menu, toggle **Show Function Tree**

- From the Function Call menu, toggle **Show Function Tree**.



Function	Call Count	Duration (ms)
—	13	47
	135	3199
:	69	62
!	747	666
=	237	331
.C	13	93
[150	1247
[.AsIs	3	47
[.data.frame	3	125
[.default	3	0
[[48	403
*	3	421
&	12	78
&&	348	4904

Figure 3.19: *Function Calls* view, table display.

Function Calls menu

The **Function Calls** view menu displays the following options:

Table 3.8: *Function Calls* view menu options.

Menu option	Description
Show Function Tree	Toggle to show the function calls in a tree view or in a tabular view. The tabular format displays the total number of calls and total duration of each function.
Refresh Function Calls	Forces an update of the function call tree or table.
Reset Function Calls	Clears the function call tree or table.

Allocations view

Displays the number of allocations the engine has performed. It breaks the allocations down into bytes and the basic S-PLUS data types.

Allocations view menu

The **Allocations** view menu displays options to refresh or reset the view, similar to the **Function Calls** view options. See Table 3.8 for more information.

S-PLUS WORKBENCH TASKS

4

Introduction	100
S-PLUS Workbench Projects	101
Setting the Workspace	101
Creating a Project	102
Setting the S-PLUS Workbench Preferences	110
Customized Perspective Views	120
Working Projects and Databases	123
Setting the Working Project	123
Changing Attached Databases	125
S-PLUS Project Files and Views	128
Creating a Script	128
Editing Code in the Script Editor	129
Running Code	134
Closing and Reopening the Project	138
S-PLUS Workbench Debugger Tasks	139
Kahan Example	139
Opening the Debug Perspective	139
Launching the debugger	140
Setting breakpoints	141
Starting execution	143
Examining the call stack	144
Examining Variables and Expressions	145
Setting a Watch Expression	146
Stepping into, over, and out of a function	148
Examining Resource Usage	150
Examining Function Calls	150

INTRODUCTION

This chapter provides the basic tasks that demonstrate using the S-PLUS Workbench. For information about basic Eclipse IDE tasks, see the Eclipse *Workbench User Guide*.

This chapter includes:

General S-PLUS Workbench tasks, including:

- Setting the Workspace, page 101
- Creating a Project, page 102
- Setting the S-PLUS Workbench Preferences, page 110
- Customized Perspective Views, page 120
- Specifying Working Projects and Databases, page 123
- Working with S-PLUS Project Files and Views, page 128

This chapter also includes tasks that introduce you to using the views and features in the S-PLUS perspective. For more information, see the section S-PLUS Workbench Projects on page 101.

Finally, this chapter includes tasks that introduce you to using the views and features in the Debug perspective. For more information, see the section S-PLUS Workbench Debugger Tasks on page 139.

S-PLUS WORKBENCH PROJECTS

Before you begin working with files in the S-PLUS Workbench, you must set your workspace and then create a project.

Setting the Workspace

When you first launch the S-PLUS Workbench, you are prompted to supply the path to your S-PLUS workspace.

To set the workspace

1. In the **Workspace Launcher** dialog (Figure 4.1), specify the directory location where the workspace **.Data** and **.metadata** databases will be stored.
2. Indicate whether you want to be prompted in future sessions to identify a workspace using this dialog.

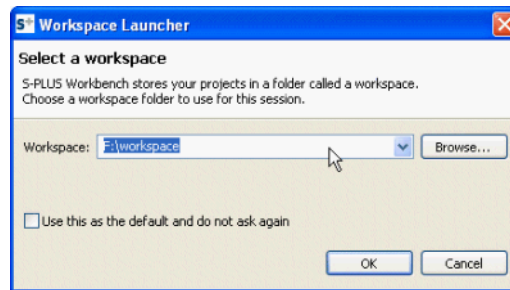


Figure 4.1: *The Workspace Launcher dialog.*

Changing the Workspace

You can switch to another workspace from within the S-PLUS Workbench user interface.

To open a different workspace in S-PLUS Workbench

1. Save your work.
1. Click **File ► Switch Workspace**.

2. In the **Workspace Launcher** dialog, provide the new workspace location.

Note

When you switch workspaces during an S-PLUS Workbench session, the current session closes, and a new session of S-PLUS Workbench starts, using the new workspace location.

After you set the workspace, create the project.

Note

On Microsoft Vista™, you must be elevated to the role of administrator to specify the default directory as **C:\Program Files\Insightful\splus80\users\yourname**; however, it is not recommended that you use this directory

Creating a Project

The S-PLUS Workbench project is a resource containing scripts and other associated files. You can use the project to control build, version, sharing, and resource management.

Understanding Project Options

Before you create a new project, consider the following scenarios, and then review the S-PLUS Workbench options.

Table 4.1: S-PLUS Workbench project scenarios.

Scenario	S-PLUS Workbench Option
You are starting an empty project with no existing files. (Note: This is the only way to create a project that is stored in your workspace.)	In the New Project wizard, specify a project name and accept the default project directory location. Your project is created as a subdirectory in the workspace directory. (The Navigators view displays the .project resource but no existing project files.)

Table 4.1: *S-PLUS Workbench project scenarios. (Continued)*

Scenario	S-PLUS Workbench Option
You have one or more project(s), and you want to work with the files at their existing location.	<p>In the New Project wizard, specify a project name, clear the Use default check box, and then browse to the location of the project files. S-PLUS Workbench works with the files at the specified location. (The Navigators view displays the .project resource and all files in the project directory.)</p> <p>Note that the project cannot overlap other projects and cannot be located under your workspace.</p>
You have an existing project, and you want to copy selected files to a workspace directory (for example, in the cases where the files are kept at a remote location, are read-only, or where you do not want to work with the original files).	<p>In the New Project wizard, specify a project name and accept the default project directory location. An empty project subdirectory is created in the workspace directory. You can then import your project files. See the section Importing Files on page 104 for more information.</p>

Based on the scenario that applies to your project needs, In the following sections, create an empty project, and then import the Census project files (the third scenario described above).

To create the example census project

1. Click **File ► New ► Project**.
2. In the **New Project** dialog, select S-PLUS Project. Click **Next**.
3. Provide the friendly project name, “Census.”
4. Accept the option **Use default**. This option creates the project directory in the default workspace location.

5. Click **Finish** to create the project.

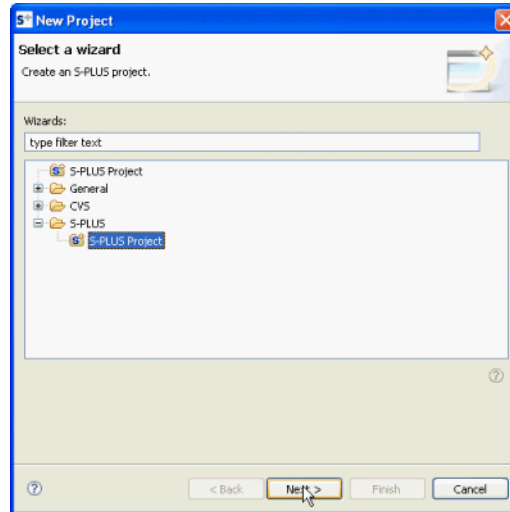


Figure 4.2: *New Project dialog.*

Note

When you create a project, you see in the **Navigator** view the **.project** resource. This resource is created by Eclipse and contains information that Eclipse uses to manage your project. You should not edit this file.

Importing Files

In this exercise, *import* the Census example, one of the examples provided with S-PLUS.

To import files

1. With the Census project selected in the **Navigator** view, click **File ► Import**.
2. In the **Import Select** dialog, select **File system**, and then click **Next**.
3. In the **Import File system** dialog, browse to the location of the census project (by default, in your installation directory at **\$HOME/samples/bigdata/census**.)
4. Select the directory, and then click **OK**. The directory name appears in the left pane, and all of the project's files appear in the right pane.

5. Select the folder name in the left pane to select all files, and then click **Finish** to add the files to your project.

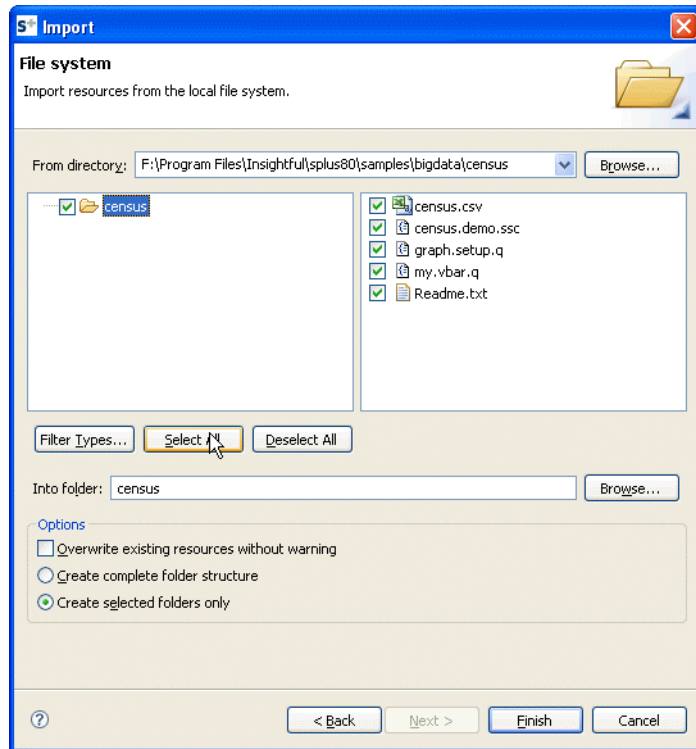


Figure 4.3: *Import File System* dialog for *Census* project.

Hint

You can select just the **.ssc** file to import if you prefer, because the script itself references the data in these files. For the purposes of this part of the exercise, we import all files.

Figure 4.4 shows the **Navigator** with the Census project and all its files.

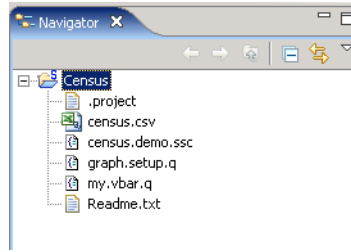


Figure 4.4: *Navigator* showing *Census* project.

Note

Alternatively, you can copy files from a different location to your project directory in your workspace. If you simply copy files, you must refresh the **Navigator** view to include the files in your project and display them in the project file list. To refresh the view, right-click the project name, and from the menu, click **Refresh**.

Loading a Library You work with S-PLUS code in the Workbench the same way you work with it in other environments, such as the Java GUI, the command line, or the Windows GUI. To load a library, in the **Console** view, simply call:

```
library(libraryname)
```

Where *libraryname* is the library to load.

For example, if you are working with S-PLUS packages, before you get started, load the `pkgutils` library:

```
library(pkgutils)
```

Adding a Second Project

In this exercise, create a project with the files for the **Boston Housing** example at their *existing* location (the second scenario described above), rather than importing the files into a workspace directory. **Boston Housing** is an example provided in the S-PLUS sample files, by default, in your installation directory at **\$HOME/samples/bigdata/boston**.

To add a project

1. Click **File ► New ► Project**.

2. In the **New Project** wizard, select **S-PLUS Project**, and then click **Next**.
3. In the **Project name** text box, type “Boston Housing,” and then clear the **Use default** check box.
4. Browse to the location of the Boston Housing sample directory, by default in the **\$HOME/samples/bigdata** directory of your S-PLUS installation. Select the **boston** directory, and then click **OK**. Click **Finish** to add the project.

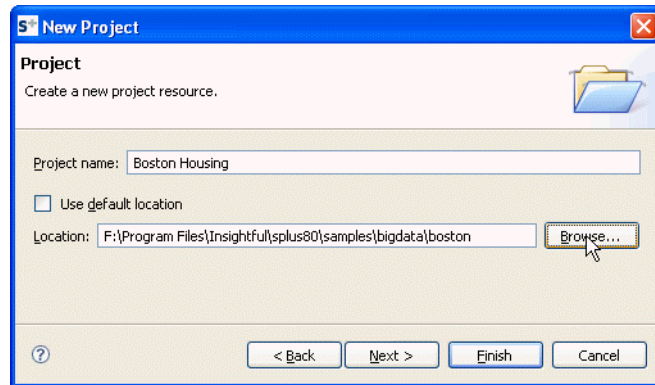


Figure 4.5: *New Project* dialog, using boston files at their installed location.

5. In the **Navigator** view, the **Boston Housing** project appears. This directory contains all of the files in that sample directory location.

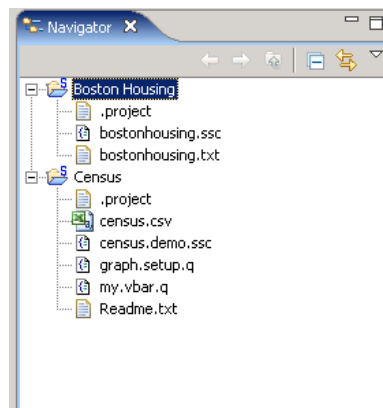


Figure 4.6: *Navigator* containing two projects.

6. You won't be using this project for the remainder of this tutorial section, so right-click the directory, and then select **Delete**.
7. In the **Confirm Delete Project** dialog, select **Do not delete contents**. (Otherwise, you will delete the sample from your installation directory.)
8. Click **Yes** to remove the project.

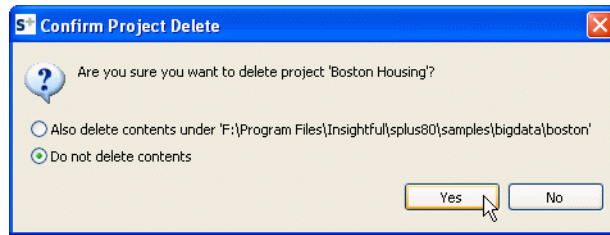


Figure 4.7: *Confirm Project Delete* dialog--do not delete the contents.

Copying Files Between Projects

You can copy files from an existing project to a working project by copying their **.ssc** files from the original project to the working project's directory. Note that to see these files in your project, you must refresh the view. To refresh the **Navigators** view, right-click the project, and from the menu, click **Refresh**. (Restarting the S-PLUS Workbench does not automatically refresh the view.) Alternatively, you can use the **File ► Import** menu command: Specify a file system, browse to the original location of the desired file, and then select only that file to import. (Importing a file into a project from another location copies that file to the project folder in your workspace.)

Adding the Sample Debugging Project

In this exercise, add another project, importing the sample files, as you did in the section To create the example census project on page 103. This second project is the project you will use later in this chapter to practice debugging tasks.

Follow the directions for creating a project on pages 103 to page 106, but instead of importing the Census project, import the kahanSum project, located in your installation directory at **\$HOME/samples/kahanSum**.

To add the kahanSum project

1. Click **File ► New ► Project**.

2. In the **New Project** dialog, select **S-PLUS Project**. Click **Next**.
3. Provide the friendly project name, “kahanSum.”
4. Accept the option **Use default**. This option creates the project directory in the default workspace location.
5. Click **Finish** to create the project.
6. With the **kahanSum** project selected in the **Navigator** view, click **File ► Import**.
7. In the **Import Select** dialog, select **File system**, and then click **Next**.
8. In the **Import File system** dialog, browse to the location of the census project (by default, in your installation directory at **/samples/kahanSum.**)
9. Select the directory, and then click **OK**. The directory name appears in the left pane, and all of the project’s files appear in the right pane.
10. Click **Select All**, and then click **Finish** to add the files to your project.

Figure 4.8 shows the **Navigator** with the kahanSum project and all its files added to the workspace. (You will work with this project later in this chapter.)

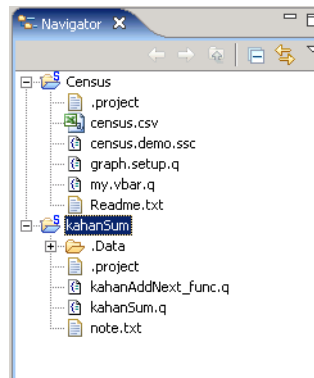


Figure 4.8: *Navigator* showing kahanSum project added to the workspace.

Setting the S-PLUS Workbench Preferences

S-PLUS provides customizations to the Eclipse IDE to accommodate the specific needs of the S-PLUS programmer. You can change the IDE to suit your development style, including adding, removing, and repositioning the views, and setting the preferences.

- To review the preference options in the **Preferences** dialog, see the section Examining S-PLUS Preferences on page 13.
- To review the views available in the S-PLUS Workbench, see the section Examining the S-PLUS Workbench GUI on page 24.
- To learn more about customizing the views in the S-PLUS Workbench, see the section Customized Perspective Views on page 120.

General Options

This section demonstrates setting specific preferences in the **Preferences** dialog.

To set text editor options

1. On the **Window** menu, click **Preferences**.
2. In the **Preferences** dialog, select **General**, and then click **Editors ► Text Editors**.
3. Review the options, including tab width (by default 4), line numbers (by default displayed), and appearance color options (by default, the system colors). You can set additional options in the **S-PLUS ► Editor** options dialog. See Figure 4.13 for an example.

To examine file association preferences

4. In the **Preferences** dialog, select **General**, and then click **Editors**. Examine the dialog pages.

5. Click **File Associations** and review the file types that the Script Editor recognizes.

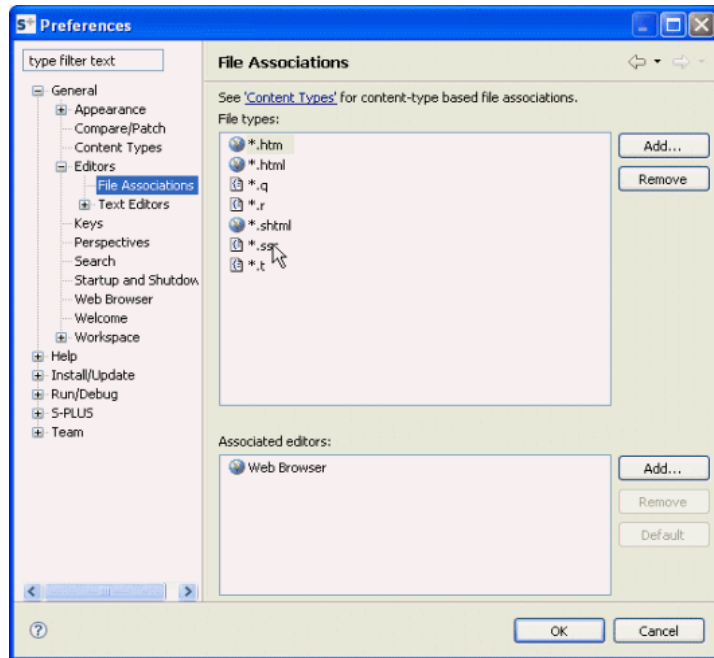


Figure 4.9: *File Associations* page.

S-PLUS View Preferences

The previous section demonstrated setting Eclipse general preferences that the S-PLUS Workbench takes advantage of. The following sections demonstrate setting preferences specific to the S-PLUS Workbench views. These preferences include general S-PLUS preferences, preferences for the **Console** view and the **Output** view, preferences for the **Output** view, and preferences for defining task tags.

To set the S-PLUS preferences

1. Click S-PLUS.

2. Review the options. Make sure the `bigdata` library loads on startup: check the check box **Run code on startup**. (The Census example demonstrated in this chapter uses the `bigdata` library.)

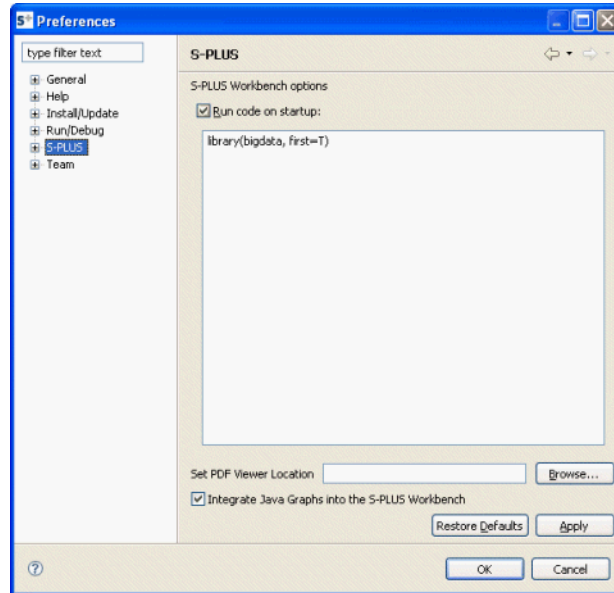


Figure 4.10: *S-PLUS Preferences* page.

To store the console history between sessions

1. In the left pane tree view, click **S-PLUS** to expand, and then click **Console** to display that page.
2. In the **Console** page, select **Store Console History between sessions**. You can use this setting to persist the contents of the **History** view to use later in the **Console** view. For more information about storing the console history and using it in the output, see the section Console Options on page 17.

- Optionally, change the input and/or output color or font to a color or font of your choice. For more information about these options, see the section Font Settings on page 17.

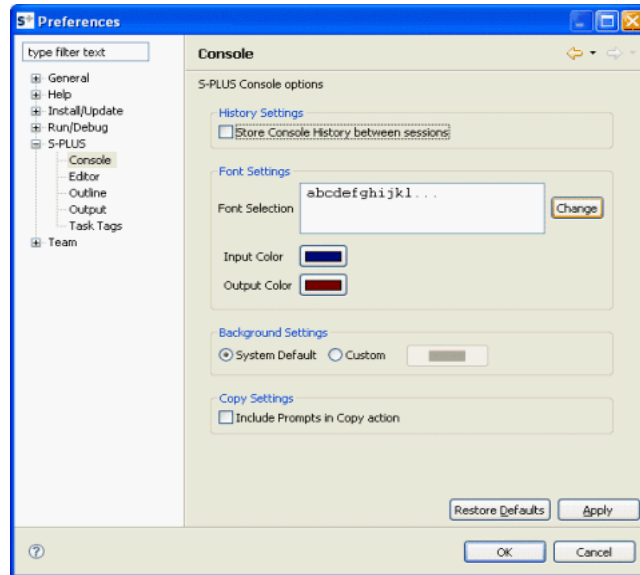


Figure 4.11: *Console* page.

To add text with a user-defined highlight color

- In the left pane tree view, under **S-PLUS**, click **Editor** to display that page.
- In the **Editor** dialog, in the **Syntax Highlighting** list box, select **User**, and then click **Choose Color**.
- In the **Color** dialog, select a color, and then click **OK**.
- In the **User Tokens** area, click **New**.

5. In the **Add Desired S-PLUS Text** dialog, select **Comma Separated Text**. In the text box, type **census**, and then click **OK**.

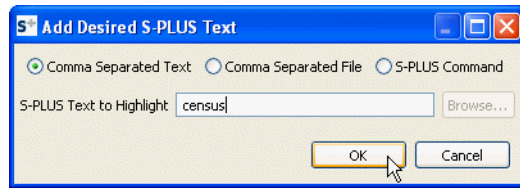


Figure 4.12: *Add Desired S-PLUS Text* dialog.

6. Note that **census** appears in the **User Tokens** list box. Click **Apply**. In later exercises, when you manipulate the **Census** project, you will see the string you selected highlighted in the color you specified. You can add other user-defined terms, including S-PLUS commands or the contents of a comma-

separated file and see how it makes tracking these items through your code easier. For more information about this option, see the section Syntax Highlighting on page 19.

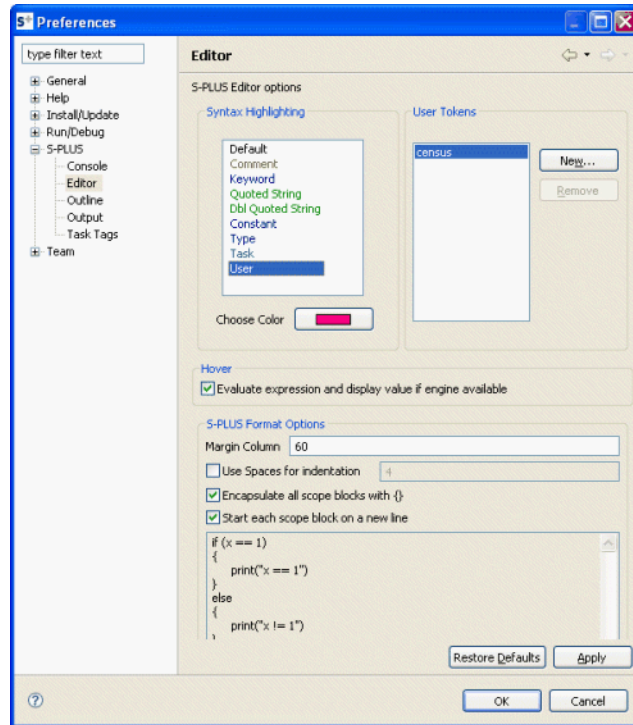


Figure 4.13: *Editor* page with *census* added as highlighted user text.

To add the contents of a comma-separated file

1. Open a text editor, such as Notepad.
2. Type some terms to highlight, separated by commas. For example, if you want to highlight in your code every time the data viewer or a graph opens, type `bd.data.viewer`, `hist`, `xyplot`, `bwplot`, `histogram`, and so on.
3. Save the file with a convenient name and to a convenient location (for example, **C:\terms.txt**).
4. Return to the S-PLUS Workbench Editor preferences dialog, and, in the **User Tokens** area, click **New**.
5. In the **Add Desired S-PLUS Text** dialog, select **Comma Separated File**.

6. Either type the file path, or click **Browse** and browse to the file location.
7. Click **OK**, and notice that all of the terms in the file are added to the **User Tokens** list. Figure 4.14 shows the **User Tokens** list with the terms added from the file.

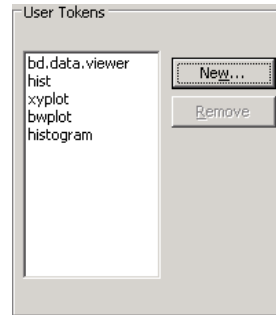


Figure 4.14: *User Tokens* list displaying the contents of a file.

To add an S-PLUS command to the User Token list

1. In the **User Tokens** area, click **New**.
2. In the **Add Desired S-PLUS Text** dialog, select **S-PLUS Command**.
3. In the **S-PLUS Text to Highlight** box, type the S-PLUS command `objects()`.
4. Click **OK**, and notice that all of the objects in the working project are added to the **User Tokens** list. Figure 4.15 shows the updated **User Tokens** list.

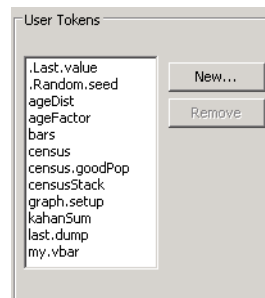


Figure 4.15: *User Tokens* list displaying working project objects.

To remove items from the **User Tokens** list, select them and click **Remove**.

To change the code formatting options

1. In the S-PLUS **Editor** options page, review the S-PLUS Format Options group.
2. Select **Use spaces for indentation**, and notice how the example display changes to reflect the default 4. Clear this option, if you choose, or change the default to add more or fewer indentation spaces.
3. Change some of the other formatting options to suit your programming style, and then click **Apply** to apply any changes to the editor.

To add a function to watch

1. In the left pane tree view, click **Outline** to display that page.
2. Click **New**.
3. In the **Add New Function to Watch** dialog, add `set.seed`. Click **OK**.

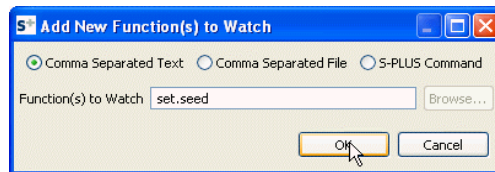


Figure 4.16: *Add New Function to Watch dialog with `set.seed`.*

4. Review the list in the **Functions to Watch** dialog. Note that `set.seed` has been added to the list. (Later, when you are working with a project that uses the `set.seed` function, you

can see its display in the **Outline** view has a special icon.) For more information about this option, see the section Functions to Watch on page 21.

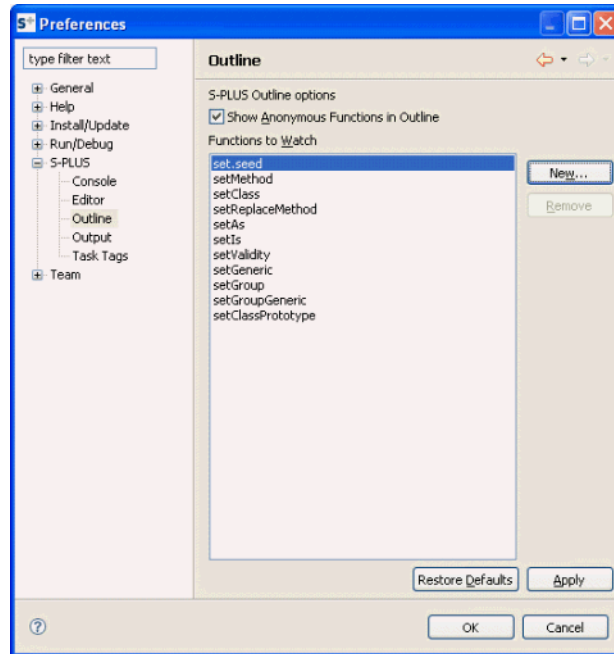


Figure 4.17: *Outline* page with `set.seed` added.

To add a task to the Task Tags options

1. In the left-pane tree view, click **Task Tags**.
2. Click **New** to display the **Add New Task Type** dialog.
3. In the **Task Name** box, type a name for a new task to watch. Set the severity to your preference, and click **OK** to add the task.

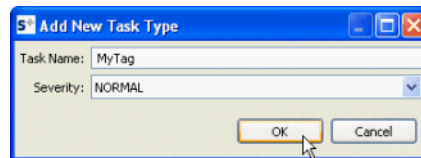


Figure 4.18: *Add New Task Type* dialog.

4. Highlight the items to change in the **S-PLUS Task Options** text box, or, using the **New**, **Remove**, **Up**, and **Down** buttons, edit the available tasks. In the Script Editor, when you type this term, prefaced with a comment character (#), the line is added to the **Tasks** view with the severity you indicate for the custom tag.

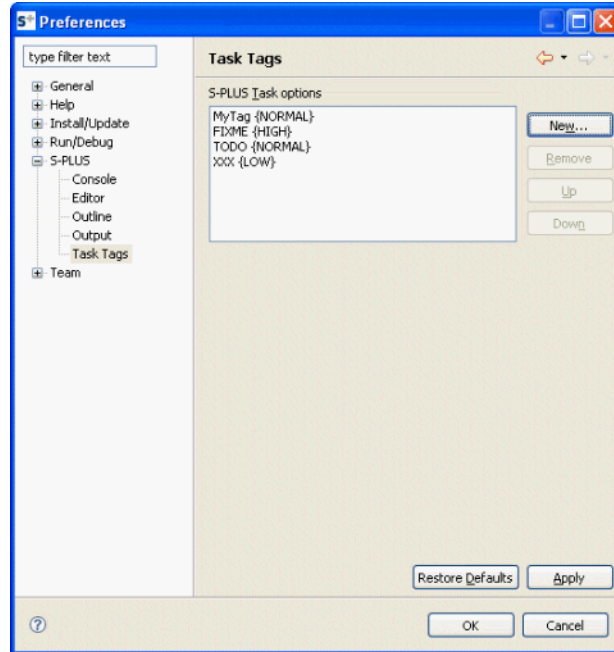


Figure 4.19: *Task Tags* page with a new task added.

5. Click **OK** or **Apply** to save your changes, or click **Restore Defaults** to return the task options to their default state.
6. Click **OK** to save your changes.

CUSTOMIZED PERSPECTIVE VIEWS

The default layout of the S-PLUS perspective presents the **Navigator** view, **Outline** view, and **History** view on the left side of the window. The **Console** view, **Objects** view, **Search Path** view, **Output** view, **Tasks** view, and **Problems** view are tiled across the bottom of the window. The Script Editor pane is empty.

To customize the S-PLUS perspective default perspective

1. Click the **Outline** view tab and drag the view beside the **Navigator** view. The **Outline** view now tiles with the **Navigator** view.
2. Click the **History** view tab and drag the view to the right; it now tiles with the other views.
3. Right-click the **Tasks** view tab and select **Fast View**. The **Tasks** view minimizes and appears as an icon in the window's status bar.
4. Click the **Output** view tab to select it.

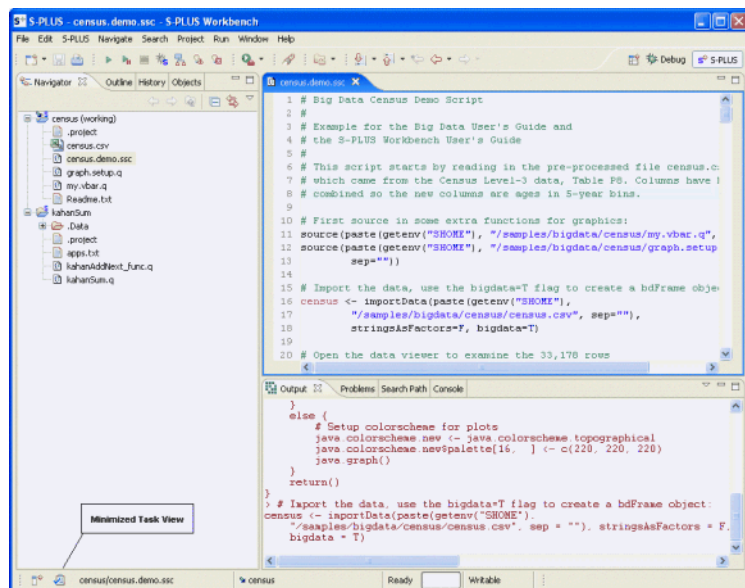


Figure 4.20: Customized S-PLUS perspective.

5. Click **Window ► Save Perspective As**.

6. In the **Name** box, type “Sample Exercise,” and then click **OK**.

The **Sample Exercise** perspective button appears on the toolbar:

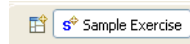


Figure 4.21: *Sample exercise perspective button.*

To change the displayed views

1. To change the views, or to display the list of available views, on the menu, click **Window ► Show View**.
2. From the submenu, select the view to display.

Alternatively, if you do not see the view you want to display, from the **Show View** menu, click **Other**, and then select a view from the **Show View** dialog. For example, if you want to display a view that is typically in the **Debug** perspective, expand **Debug**, and then select a view from the list.

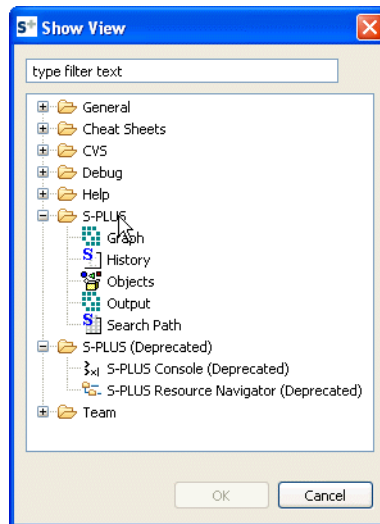


Figure 4.22: *Show View dialog.*

- If the view is not currently visible in the UI, selecting it displays the view and gives it focus in the UI.

- If the view is available, selecting it gives it focus in the UI.

Note

The **S-PLUS (Deprecated)** folder contains options to display S-PLUS views that are deprecated in your current release. You can use these views; however, in the future, they will be unavailable.

Because of ongoing improvements and implementation changes, occasionally S-PLUS views might change or become obsolete. If you use a workspace containing a deprecated view (that is, created prior to the release in which it was deprecated), you see the (Deprecated) label in the view's tab, and the view is moved to the S-PLUS (Deprecated) folder in the view management tool.

You can change your workspace to use the newer view. To change to a new view permanently:

- Reset the perspective using the **Windows ► Reset Perspective** menu option; or
- Delete the following file from your workspace directory:
<WORKSPACE>/.metadata/.plugins/org.eclipse.ui.workbench/workbench.xml.

To return to the S-PLUS perspective default

1. Click the perspective button to the left of the **Sample Exercise** button, and then click **Other**.
2. In the **Select Perspective** dialog, select **S-PLUS (default)**, and then click **OK**. The perspective returns to its previous layout.

You can select other views to display in your perspective.

WORKING PROJECTS AND DATABASES

This section describes setting working projects and changing databases.

The S-PLUS Workbench provides the following ways you can store your data objects:

- In the working project **.Data**, where the objects are available only to the project.
- In the workspace **.Data**, where the objects are available to all projects in the workspace.

You can change the **.Data** storage option at any time by setting any project in the workspace as the working project, or toggling off the working project option and writing data objects to the workspace **.Data** database.

Setting the Working Project

When you create a workspace, a **.Data** database is created in the workspace, and (after you refresh the view) the workspace path appears in the first position in the **Search Path** view, as shown in Figure 4.23. If you specify no working project, the S-PLUS Workbench writes data objects to the workspace **.Data** database, and the objects in that **.Data** database are available to all projects in the workspace.

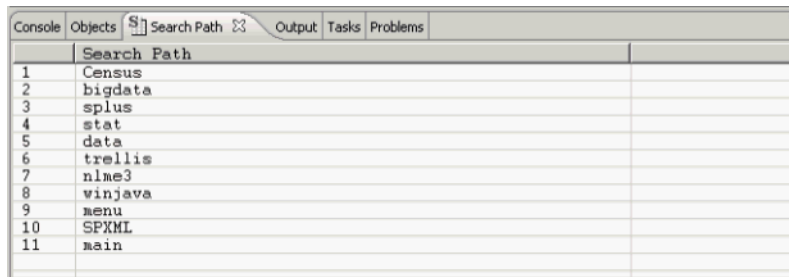


Figure 4.23: *Search Path* view with first position set to the workspace.

When you create a project and import project files, the S-PLUS Workbench creates a **.Data** in the project, sets it as the working project, and sets the project in the first position in the **Search Path** view. Any objects created are added to the working project's **.Data** database.

To set the working project

1. Select the project to set as the working project.
2. From the main menu, click **File ► Toggle Working S-PLUS Project**. (Alternatively, in the **Navigator**, right-click the project that you want to set as the working project, and from the context sensitive menu, click **Toggle Working S-PLUS Project**.) Figure 4.24 shows the context menu.

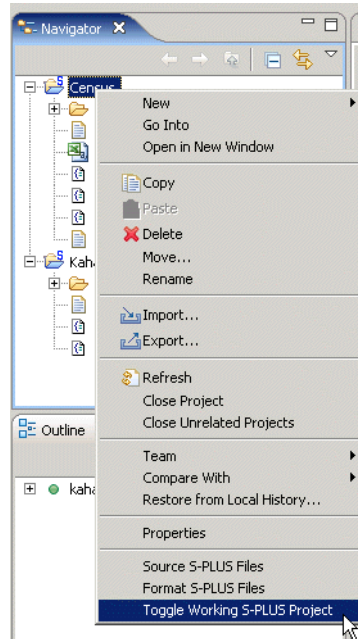


Figure 4.24: *Toggle Working S-PLUS Project on the Navigator context-sensitive menu.*

The selected project is displayed as the working project. Any objects you create are stored in the **Census .Data** database until you choose another project as the working project, or toggle off the working project, so the workspace **.Data** is the database.

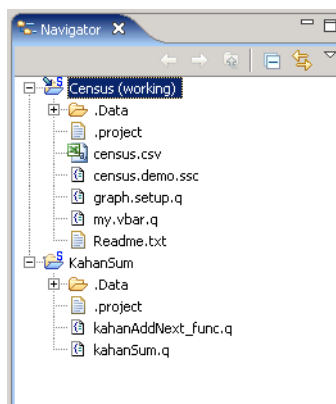


Figure 4.25: *Census set as the working project.*

Changing Attached Databases

S-PLUS recognizes libraries, modules, lists, and directories as legitimate object databases. You can add and detach any of these types of databases to the **Search Path** view.

By default, the **Search Path** view displays the full path of the working database and all of the attached S-PLUS data libraries. Objects existing in a recognized active database appear in the **Objects** view.

Adding a Database

Objects in an added database appear in **Objects** view when you refresh the view to that database. See the section Examining Objects on page 132.

To add a library

1. Right-click the **Search Path** view.
2. From the right-click menu, click **Add Library**.
3. In the **Attach Library** dialog, type MASS. Clear the **Attach at top of search list** check box to indicate that you want add the library to the bottom position.

4. Click **OK** and examine the **Search Path** view for the change.

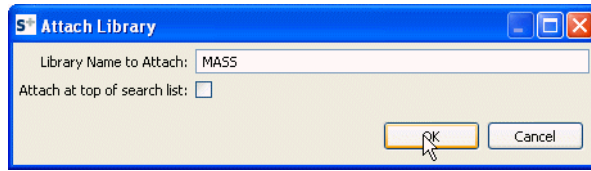


Figure 4.26: *Attach Library* dialog.

To add a module

1. From the right-click **Search Path** view menu, click **Add Module**.
2. In the **Attach Module** dialog, provide an installed module name and indicate whether to add it to the first position.
3. Click **OK** and examine the **Search Path** view for the change.

To add a directory

1. Right-click the **Search Path** view.
2. From the menu, click **Attach Directory**.
3. In the **Attach** dialog, in the **Directory to attach** text box, browse to the directory location.
4. In the **Label** text box, type Projects
5. In the **Position** text box, type 4.
6. Click **OK** and examine the **Search Path** view. The label you provided should appear at position 4.

Detaching a Database

From the **Search Path** view, you can detach a database from your current session.

To detach a database

1. In the **Search Path** view, right-click **bigdata**.
2. In the right-click menu, select **Detach**.
3. Examine the **Search Path** view. The Big Data library is no longer attached.

Refreshing the View

When you refresh the view, any changes to the **Search Path** view that have not been reflected in a recent change are displayed. For example, if you add a library by calling the load function in an S-PLUS script, the change is not immediately displayed in the **Search Path** view.

To refresh the view

1. Using the **Console** view, reattach the Big Data library. In the **Console** view, type

```
library(bigdata, first = T)
```
2. Right-click the **Search Path** view.
3. In the right-click menu, click **Refresh**. Notice that the Big Data library appears as attached in the first position (position 2).

S-PLUS PROJECT FILES AND VIEWS

The S-PLUS Workbench recognizes *.ssc, *.q, *.r, and *.t files, all file extensions common in S-PLUS code.

Creating a Script

You can create a new S-PLUS script file, or you can import an existing script file. The following two examples demonstrate both techniques.

To create a new script file

1. Click **File ► New ► Other**.
2. In the **Select a wizard** dialog, select **S-PLUS Script**. Click **Next**.
3. In the **New File** dialog, select the parent directory (the Census project directory)
4. In the **File name** text box, type **Sample.ssc**.
5. Click **Finish** to create the file.

We won't work with this file for this exercise, so you can either disregard the file, or delete it from your project. Alternatively, you can open the file, add some S-PLUS code, and save it in the project.

Viewing Project Files

The **Navigator** view displays the project files. In Windows, if you have Microsoft Excel installed, you can open a CSV file in an external window. In this project, only the files identified in **Windows ► Preferences** in the File Extensions page open in the Script editor.

Removing files from a project

Because the project script imports the data in the files from their installation directory in S-PLUS, you don't need to have them all in the project. However, removing an imported file deletes it from your project directory, so remove individual files with care.

To remove a file from the Census project

1. In the **Navigator** view, open the Census project and select all files except the **.project** file and **census.demo.ssc**.
2. Right-click the selected files, and then click **Delete**.

3. In the **Confirm Multiple Resource Delete** dialog, click **Yes** to remove the files from the project. The **Navigator** view should now just display the Census Project directory, the project file, and **census.demo.ssc**:

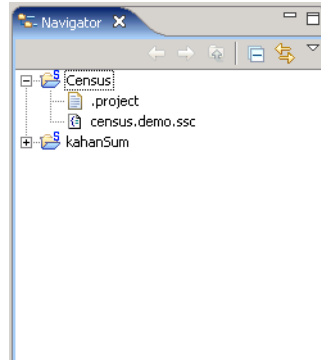


Figure 4.27: *Navigator* view after deleting files

Editing Code in the Script Editor

The S-PLUS script is a text file that you can edit in the Script Editor. In this exercise, just edit **census.demo.ssc** using the menu items provided specifically for S-PLUS.

To edit script code

1. In the **Navigator** view, double-click the file **census.demo.ssc** to open it in the Script Editor and examine the script. Note that:
 - The comment text appears in the Script Editor as green. (You can change this default color in the **Preferences** dialog. See the *Eclipse Workbench User Guide* and the section Setting the S-PLUS Workbench Preferences on page 110 for more information.)
 - Note that the term `census` appears in the color you specified in the section To add text with a user-defined highlight color on page 113.
 - The line that has focus appears highlighted.
 - The line numbers appear to the left of the script text.
2. Scroll to line 17 and highlight the line and the next line:

```
"/samples/bigdata/census/census.csv", sep=""),  
stringsAsFactors=F, bigdata=T)
```

3. Click **S-PLUS ► Shift Left**. The code shifts to the left.
4. Click **S-PLUS ► Format**. This command formats the entire script. Note that the formatting change you made in the previous step has been reverted. Also note that the line numbers for formatted functions are highlighted.

Hint

The line numbers for any line changed in your script are highlighted until the next time you save your work.

5. Scroll to the line containing the code
`graph.setup(Name="USA")`
6. Click **S-PLUS ► Toggle Comment** to add a comment character. Notice that the script text color changes to indicate that the line is no longer a comment.
7. Repeat step 6, or type `CTRL+SHIFT+#` to remove the comment.

To edit a function definition

1. In the Script Editor, select the function whose definition you want to edit.
2. Press the ctrl key and click the function again.
3. The function definition opens in a temporary file in the Script Editor.

Alternatively, you can right-click a function name, and from the menu, click **Find**. **Find** searches files currently open in the Script Editor, then files in the working project, and finally in the S-PLUS database for the function definition.

Note

Any code changes you make in an editor are not recognized by S-PLUS until you source the code.

To find all references to a function

1. Right-click the function whose references you want to find.
2. From the menu, click Find References.
3. Review the results in the **Search** view. Figure 4.28 displays the results of running **Find References** on the function `hist` in the Census project.

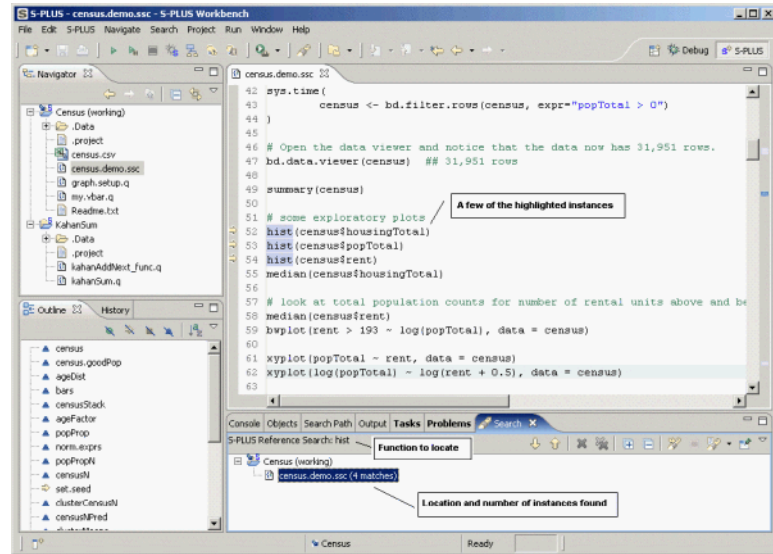


Figure 4.28: The **Search** view after running **Find References** on `hist`.

Examining the Outline

The **Outline** view displays all of the items (objects, functions, and so on) that are contained in the open script. **Outline** view is not editable.

To examine the outline

1. Examine the objects that appear in the **Outline** view. Note that `set.seed` appears with a yellow arrow next to it, because in the section Setting the S-PLUS Workbench Preferences on page 110, you indicated that `set.seed` was a function to watch.
2. Scroll through the **Outline** view list and highlight an object. Note that the Script Editor scrolls to, and highlights, the line where the object appears.

Examining Objects

When you start a new workspace, the **Objects** view is not populated.

Details about your project's objects (and all objects in your database) will appear in the **Objects** view. **Objects** view is not editable; however, you can refresh the contents, delete objects, or change the view to another attached database. To refresh the view, right-click the **Objects** view and click **Refresh**.

To examine the objects

1. Select the **Objects** view tab to display the objects and their details. By default, the objects are displayed sorted by name.
2. Right-click the **Objects** view table pane and, in the context-sensitive menu, click **bigdata**. The Big Data library objects are displayed in the **Objects** view. (It might take a few moments to display all of the objects.)
3. Re-sort the objects by any property displayed in the **Objects** view by clicking the property's column title.

To display hidden objects

1. In the **Objects** view, right-click the table pane to display the context-sensitive menu.
2. Examine the menu. Note that, by default the S-PLUS system objects are hidden.
3. On the menu, click **Hide S-PLUS System Objects** to clear the selection.
4. Examine the **Objects** view table pane and tree view pane to see the S-PLUS system objects in your project.

To select another object database

1. Right-click the **Objects** view and, in the right-click menu, click your current working directory (the directory at the top of the list). The project objects are displayed in the **Objects** view. (It might take a few seconds to display all of the objects.)

Adding a Task to A Script

The **Tasks** view displays outstanding project tasks. As discussed in the section Setting the S-PLUS Workbench Preferences on page 110, the indicators for task levels are stored in the **Preferences** dialog. (Click **Windows ► Preferences** to display them.) You can add a task in one of two ways:

- Add the task directly to the **Tasks** view.
- Add the task to the script file.

To add a task directly to the Tasks view.

1. Click the **Tasks** view tab to display its contents.
2. Right-click the view, and then click **Add Task**.
3. In the **Add Task** dialog, provide the description and priority level of the task.

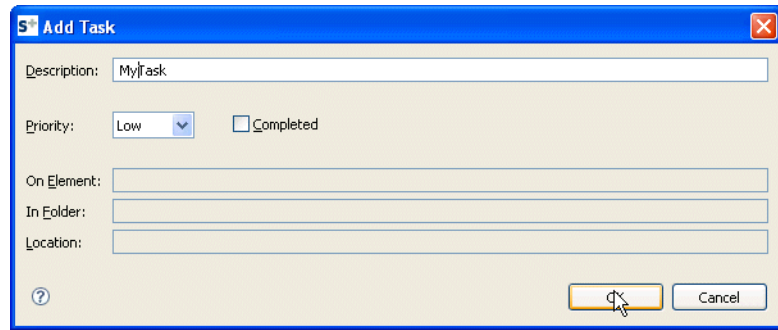


Figure 4.29: *Add Task dialog.*

4. Click **OK** to save and display the new task.

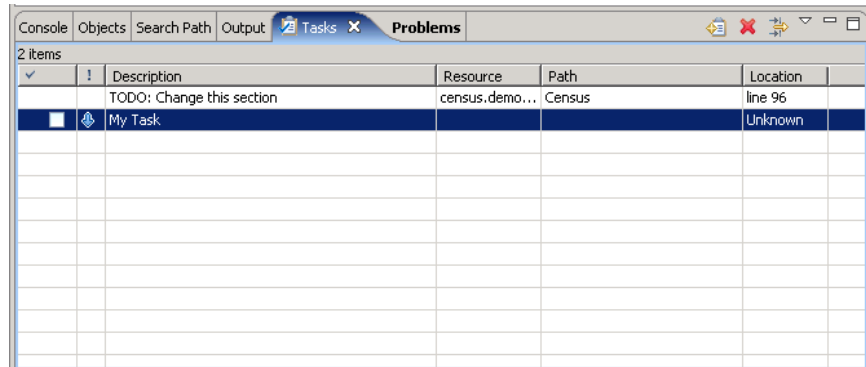


Figure 4.30: *A generic task in the Tasks view.*

A task added directly to the **Tasks** view displays a check box (for marking the task complete) in the **Tasks** view's first column. It does not display a reference to a resource, a directory, or a location.

To add a task in the script file

In the script file, select a blank line.

1. Type the following text:
#FIXME: Change this section.
2. Save the script file.

Note that the FIXME comment appears in the **Tasks** view as a high-level task, with a red exclamation mark in its second column. The task also displays information about its resource, directory, and line location. You can go directly to any task in your script by double-clicking it in the **Tasks** view.

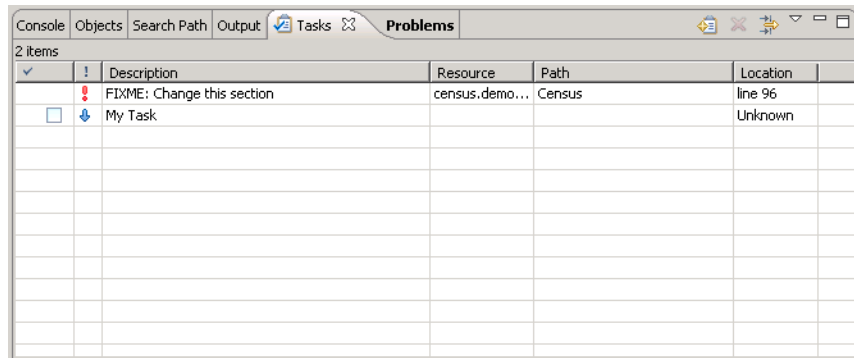


Figure 4.31: A *FIXME* task in the **Tasks** view.

3. In the Script Editor, change the level of the task by changing *FIXME* to *TODO* and save the file. Note that the exclamation mark disappears, and the task becomes a normal level task.

Running Code

You can run your S-PLUS script code directly from Eclipse in two ways, which are described in the following section.

Script Running Options

The S-PLUS Workbench provides the following customized solutions for running your scripts from either the S-PLUS perspective or the Debug perspective.

Table 4.2: *Script Editor options for running code.*

Option	Description
Copy to Console	Available from the right-click menu in the Script Editor and from the S-PLUS menu, this option copies the selected code and pastes it into the Console view. See the section Copying Script Code to the Console on page 135.
Run S-PLUS Code	Available from the Run menu, by pressing F9, on the toolbar, and from the right-click menu in the Script Editor. This option runs the selected code (or all code, if none is selected), and then displays output in the Output view. See the section Running Code and Reviewing the Output on page 137 for more information.
Run Current File	Available from the S-PLUS menu. This option runs the file that is open.
Run Next S-Plus Command	Available from the Run menu and from the S-PLUS Workbench toolbar. This option runs the currently selected S expression or, if the cursor is not exactly on an expression, the next expression.

Copying Script Code to the Console

The **Console** view is an editable view (in other words, you can type commands and run them by pressing ENTER); therefore, when you copy script contents to the **Console** view using **Copy** and **Paste** actions, you must include the line return, or the script will not run. This behavior is consistent with the S-PLUS **Commands** window, in the S-PLUS GUI, which also requires a line return to run code.

Also like the S-PLUS **Commands** window, the **Console** view concatenates the code that runs throughout your S-PLUS Workbench session, so you can review and save it.

To run copied script code

1. From within the Census project's script, select lines 1 - 13 in the script. Be sure to select the line return at the end of line 13.
2. Right-click the code and click **Copy to Console**. The selected code is copied immediately to the **Console** view and runs. You do not need to paste it in the **Console** view.
3. Repeat steps 1 and 2 for lines 15-18.
4. Finally, repeat steps 1 and 2 for a few more lines.

(You can select all of the code, but if you do so, it appears in the **History** view as one line. By following the steps above, the **History** view reflects the three different calls to run the code. See the section Examining the History view on page 136 for more information.)

Copying Script Code from the Console view

You can select and copy code from the **Console** view.

- To copy just code, select the code in the console that you want to copy, right-click the **Console** view, and from the menu, click **Copy**.
- To copy code and the prompts (> and +) in the **Console** view, set the **Window ► Preferences** option **Include Prompts in Copy action** on the **Console/Output** page. After you set this option, any lines you select and copy using the right-click **Copy** action includes both code and prompts.
- To copy the entire contents of the **Console** view, right-click the view, and on the menu, click **Select All**, and then right-click again and select **Copy**.

Examining the History view

This exercise uses the script code run in the section Copying Script Code to the Console on page 135.

The **History** view reflects the code run in the **Console** view. Note that the **History** view displays each selection you make, even if it is more than one command, on one line, and if the line extends beyond about 50 characters, the **History** view displays an ellipse (...) to indicate more code. To display each line of code in the **History** view, you must run the lines individually.

To examine the history

1. To examine and rerun code from the **History** view.
2. Click the **History** view tab to give it focus.
3. Right-click the first line of code, and click **Select input**. The code is copied to the **Console** view. You must return to the **Console** view and press ENTER to run the code.

(Alternatively, double-click the code in the **History** view to copy it to the **Console** view and run it.)

You can scroll through the individual entries in the **History** view; as you scroll, the selection appears in the **Console** view. To run a selected item, switch from the **History** view to the **Console** view and press ENTER at the end of the code line.

Running Code and Reviewing the Output

You can run code directly from the Script Editor by using the **Run S-PLUS Code** feature.

To run code

1. Select the **Output** view tab.
2. In the Script Editor, select the code to run (or, to run the whole script, select nothing), and press F9, or on the toolbar, click **RunS-PLUS Code**.

The **Output** view displays the run code and any S-PLUS messages.

To run the current expression

- On the S-PLUS toolbar, click **Run Next S-PLUS Command**. The currently-selected S expression runs, and the next expression is selected. (If the cursor location does not match an expression exactly, the next expression is evaluated.)

Fixing Problems in the Code

Introduce a programmatic problem in the script to examine the results in the **Problems** view.

To examine problems

1. In the Script Editor, on line 13 of the script, remove the closing parenthesis.

2. Save the file. Note that the **Problems** view tab shows bold text.
3. Click the **Problems** view tab to display the view.
4. Click the problem description. Note that the Script Editor highlights the line where the code is broken.
5. In the Script Editor, replace the missing parenthesis and save your file. Note that the problem disappears from the **Problems** view.

Closing and Reopening the Project

The S-PLUS Workbench maintains a list of your projects in the **Navigator** view, even after you close all associated files.

To close the project

1. Select the project to close.
2. Right-click the **Navigator** view and, from the menu, click **Close Project**.
3. Examine the **Objects** view and note that it still displays project or workspace objects.

To reopen the closed project

1. Select the project.
2. Right-click the **Navigator** view and, from the menu, click **Open Project**.

In the next section, examine the Debug perspective using a different example, creating a new project. You can close the **Census** project at this point, if you choose.

S-PLUS WORKBENCH DEBUGGER TASKS

This section describes basic tasks you will want to know how to perform on a simple file set.

Note

If you open a file using the **File ► Open File** menu command, and that file is not in an S-PLUS Workbench project, you cannot set a breakpoint in that file.

The following instruction works with the kahanSum example, located in your **\$HOME/samples/**directory. To create a kahanSum project, follow the steps to create a project (see the section Creating a Project on page 102).

Kahan Example

In numerical analysis, the Kahan summation algorithm minimizes the error when adding a sequence of finite precision floating point numbers. (It is also called compensated summation. This algorithm is attributed to William Kahan.)

The S-PLUS Debugger example uses a simple Kahan summation algorithm captured in two files. If you have not already done so, create this project and import the example files. See the section Adding the Sample Debugging Project on page 108.

The project's two files are as follows:

- **kahanSum.q** contains the function kahanSum.
- **kahanAddNext_func.q** contains the function, kahanAddNext, which is called by the function kahanSum.

Opening the Debug Perspective

Before using the Workbench Debugger and Profiler, you must open the Debug perspective. If you have closed your project in the previous exercise and want to continue practicing using the S-PLUS Debugger, first re-open your project, and open the two files in the Script Editor. Next, change to the Debug perspective.

For a more in-depth description of the Debug perspective, see the section Debug Perspective Options and Preferences on page 68.

To open the Debug perspective

1. On the perspective toolbar, click **Open Perspective**.

- From the menu, select **Debug**.

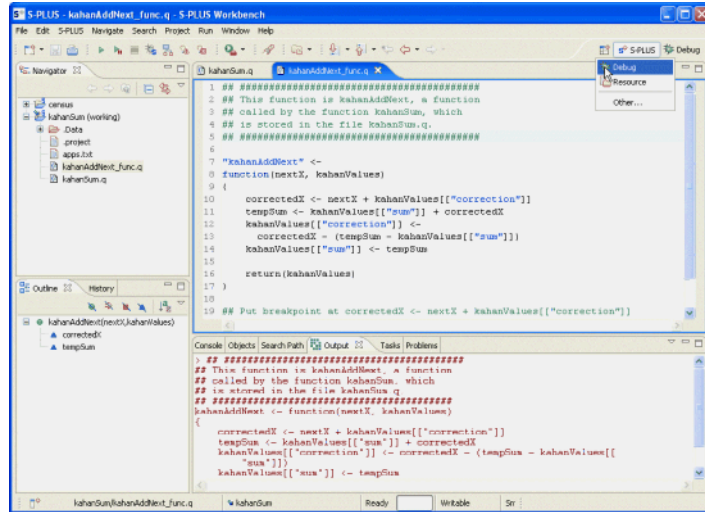


Figure 4.32: The *Open Perspective* menu options.

The **Debug** perspective button appears to the left of the **S-PLUS** perspective button, and the perspective changes to the Debug perspective as shown in Figure 4.33.

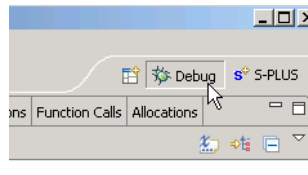


Figure 4.33: Selecting the *Debug* perspective.

Now, you can toggle between the **S-PLUS** perspective and the **Debug** perspective by clicking their respective buttons. The **Debug** perspective button stays visible in this and future S-PLUS Workbench sessions, unless you close it by right-clicking its button and clicking **Close**.

Launching the debugger

To start debugging, first activate the S-PLUS debugger using one of the following methods:

- On the toolbar, click **Toggle S-Plus Debugger** ().

- On the menu, click the **Run ► Toggle S-PLUS Debugger**.
- On the keyboard, press CTRL+ALT+D.

After you activate the debugger, any expression you type in the **Console** view, or that you run by clicking **Run S-PLUS Code** (▶) on the toolbar, invokes the debugger.

Setting breakpoints

One of the most basic debugging tasks is setting a breakpoint. Set breakpoints at locations in your code where you want to evaluate variables. In this exercise, set the breakpoints in the Script Editor. For a more in-depth description of the Script Editor, see section S-PLUS Workbench Script Editor on page 46.

To set the breakpoints

1. From the Project files, open the file **kahanAddNext_func.q** in the Script Editor. (Note that to work with the debugger, your
2. Find the line in the code that reads:

```
correctedX <- nextX + kahanValues[["correction"]].
```
3. In the left margin, right-click to display the menu, and then select **Toggle Breakpoint**. (Double-clicking the left margin next to the code line also adds the breakpoint.)
4. Open the file **kahanSum.q** in the Script Editor.
5. Find the line in the code that reads:

```
kahanValues <- kahanAddNext(x[i], kahanValues).
```

- Repeat step 3 to put a breakpoint at this line.

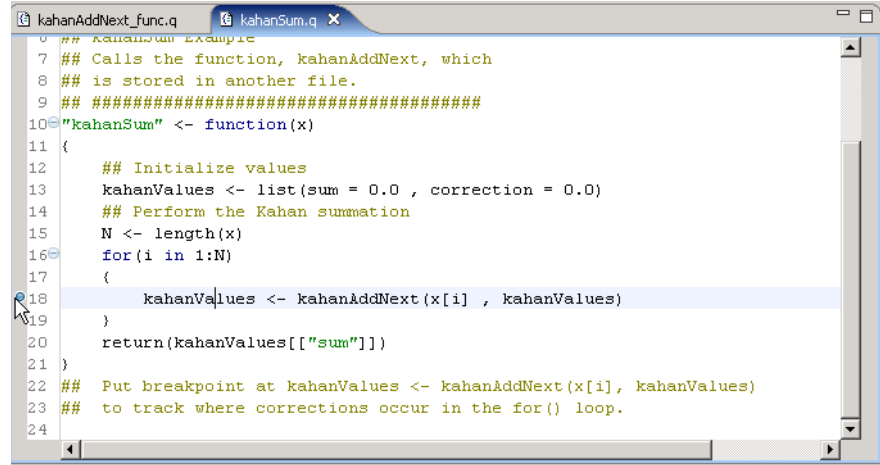


Figure 4.34: Breakpoint set in the kahanSum function.

Note

Setting breakpoints in code files the S-PLUS Workbench does not affect the file if you open it in the S-PLUS GUI in Windows. Breakpoints are evaluated only in the S-PLUS Workbench, and only when the debugger is engaged.

To examine your breakpoints

- Click the **Breakpoints** view tab. Both breakpoints you set appear in this view.

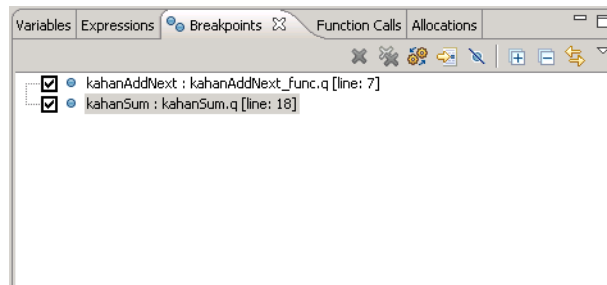



Figure 4.35: kahanSum breakpoints in **Breakpoints** view.


2. Right-click the breakpoint `kahanAddNext`, and from the context-sensitive menu, click **Go to File**. Note that the `kahanAddNext_func.q` file opens in the Script Editor, and the line with the breakpoint is highlighted.
3. In the **Breakpoints** view, clear the check box next to the `kahanAddNext` breakpoint. Note that the icon changes from a solid circle to a blank circle in both the **Breakpoints** view and in the Script file margin. This action disables the breakpoint in future sessions but does not remove it.
4. Select the check box to enable the breakpoint.
5. On the **Breakpoints** view toolbar, click **Skip All Breakpoints** () . Toggling this option disregards but maintains (that is, does not remove or disable) all breakpoints in the **Breakpoints** view.
6. Take some time manipulating the breakpoints using the menu options and buttons in the **Breakpoints** view. For a more in-depth description of the **Breakpoints** view, see section Breakpoints view on page 89.

When you have finished, re-set the breakpoints in the files as described in the section To set the breakpoints on page 141.

Starting execution


Before you run the debugger, first initialize the objects and set the output display option.

To initialize the objects

1. Open the file `kahanAddNext_func.q` in the Script Editor.
2. On the toolbar, click **Run S-PLUS Code** () .
3. Repeat steps 2 and 3 for the file `kahanSum.q`.
4. In the **Console** view, at the prompt, type the following code:

```
options(digits=17)
```

To start the debugging session

1. Engage the debugger by clicking its toolbar button () .
2. Click the **Debug** view tab to display its contents.

3. In the **Console** view, at the prompt, type the following code:
`kahanSum(rep(1000000000.1, 10))`

Examining the call stack

After you have started the debugging session, examine the UI:

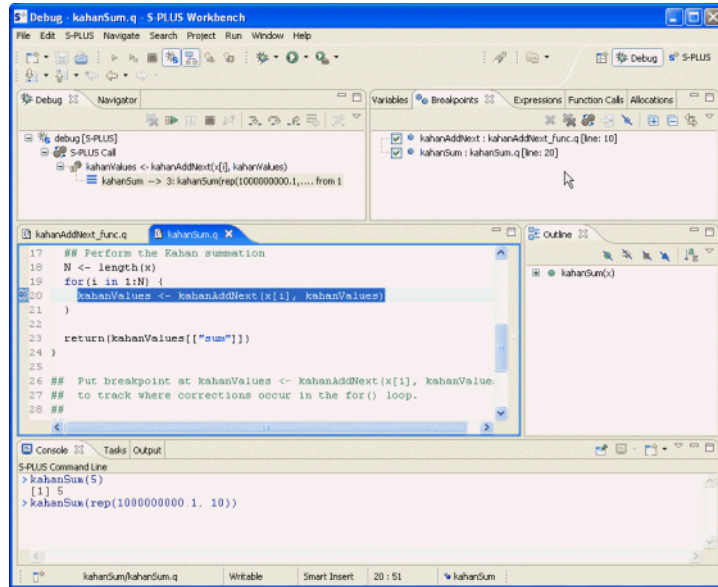


Figure 4.36: At the first breakpoint in `kahanSum`.

To examine the call stack

1. Note that the Script Editor highlights the breakpoint line.
2. Note that the **Debug** view shows the contents of the call stack.

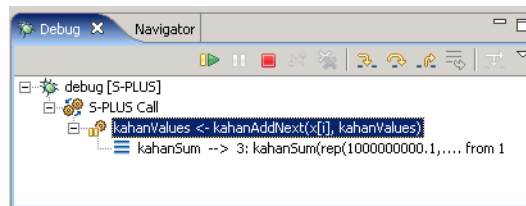


Figure 4.37: The **Debug** view containing the call stack.


To resume debugging

1. Resume debugging by clicking the **Resume** button ().

2. Re-examine the **Debug** view, and note that the debugger stops at the next break point: the first breakpoint in the `kahanAddNext` function.

For a more in-depth description of the **Debug** view, see the section *Debug view* on page 76.

To remove a breakpoint mid-session and resume debugging

1. You don't need the breakpoint in `kahanAddNext`, so remove it. In the **Breakpoints** view, select the breakpoint for `kahanAddNext`.
2. On the **Breakpoints** view toolbar, click **Remove Selected Breakpoints** .
3. Click the **Resume** button again to run the debugger to the next breakpoint. The code runs the for loop and stops at the `kahanSum` breakpoint again. Observe the results in the **Debug** view.
4. Click **Resume** a few more times to continue debugging to the first calculate correction. In the next section, examine the results of the first calculated correction.

Examining Variables and Expressions

As you debug, at each breakpoint or step, the debugger re-evaluates the variables and displays the results in the **Variables** view. At any breakpoint or stopping point, you can review, but not edit or alter, the variables at the current frame.

For a more in-depth description of the **Variables** view, see the section *Variables view* on page 82.

To examine the variables

1. Click the **Variables** view to display its contents.
2. In the **Debug** view, highlight the last line in the call stack.
3. Note that the **Variables** view displays the variables resulting from the code run so far.

4. Highlight a variable in the **Variables** view and note that its value is displayed in the **Details** pane.

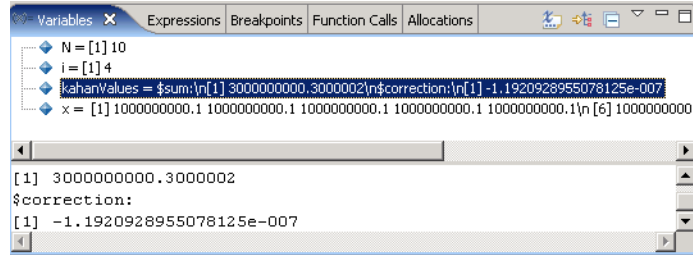



Figure 4.38: *Variables* view during debugging session.

5. On the **Variables** view toolbar, click the Show Type Names button (). Note that the **Variables** view now displays the variable type information.

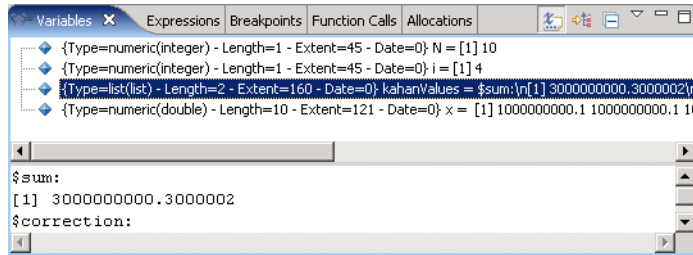


Figure 4.39: *Variables* view showing type names for the variables.

6. Take some time examining the variables using the menu options and buttons in the **Variables** view, clicking **Resume** in the **Debug** view and watching the variables change.

Setting a Watch Expression

You can track variable assignments in the **Variables** view, and you can track variables and expressions in the **Expressions** view. This section demonstrates how to track individual variable assignments and interesting expressions.

Note on Expressions

An *expression* is any syntactical interaction that S-PLUS can evaluate. Expressions persist from session to session. S-PLUS recognizes a wide variety of expressions, but in interactive use the most common are

names, which return the current definition of the named data object, and function calls, which carry out a specified computation. Any of the following are S-PLUS expressions:

```
1:10
rnorm(5)
mean(1:10)
traceback()
```

If you were debugging a function, for example:

```
incrementByTwo <- function(x) {
  * x + 2
}
```

you could have an expression that evaluated:

```
x + 2
```

at the breakpoint (denoted with the `*` in the above function definition).

For more information about expressions, see the *Programmer's Guide*, or see the S-PLUS Help topic **ExpressionLanguage**.

To watch expressions

1. Right-click the **Expressions** view to display the context-sensitive menu.
2. From the menu, select Add Watch Expression.
3. In the Add Watch Expression dialog, type the following:

```
kahanValues[["sum"]]
```

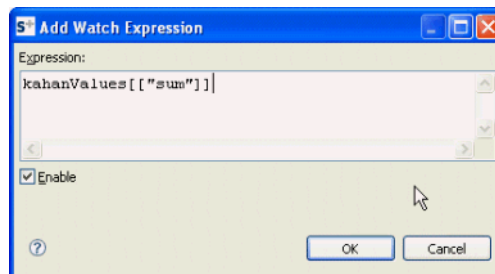


Figure 4.40: *Add Watch Expression* dialog with expression added.

4. Add a second expression to watch:

```
kahanValues[["correction"]]
```

5. Click the **Variables** view and right-click the variable `x`.
6. On the context-sensitive menu, click **Create Watch Expressions**.
7. The debugger returns to the **Expressions** view. Note that the variable `X` is now on the list.
8. Add the following two more expressions to watch:

```
sum(x)
```

```
sum(x) - kahanValues[["sum"]]
```

The first expression provides the sum of the variable `x`.

The second expression provides the difference between the sum of `x` and the sum of `kahanValues`.

Optionally, restart debugging to start evaluating the expressions from the start, pausing to see the results between the **Variables** view and the **Expressions** view.

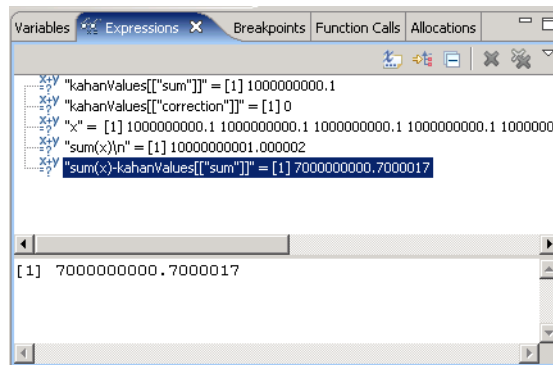



Figure 4.41: Evaluating expressions.

Stepping into, over, and out of a function

Other common debugging tasks include stepping through, over, and out of a routine's functions. When you step through a routine, the debugger pauses at every function and function body, giving you the opportunity to examine the results. This feature is also used to enter `if/else` statements, for loops, `while` loops, and other routines.

To step into a routine

1. Click the **Step Into** button () . Repeat to continue to step into routines and their bodies.
2. Note that as you click **Step Into**, the code being evaluated is highlighted in the Script Editor and is evaluated in the call stack.
3. Keep stepping into the code until you come to an internal function. (See Figure 4.42).

Because this function is not in your project files, note that the function is displayed in a temporary file. (You can set a breakpoint in such a function, and continue to evaluate it in future debugging sessions.)

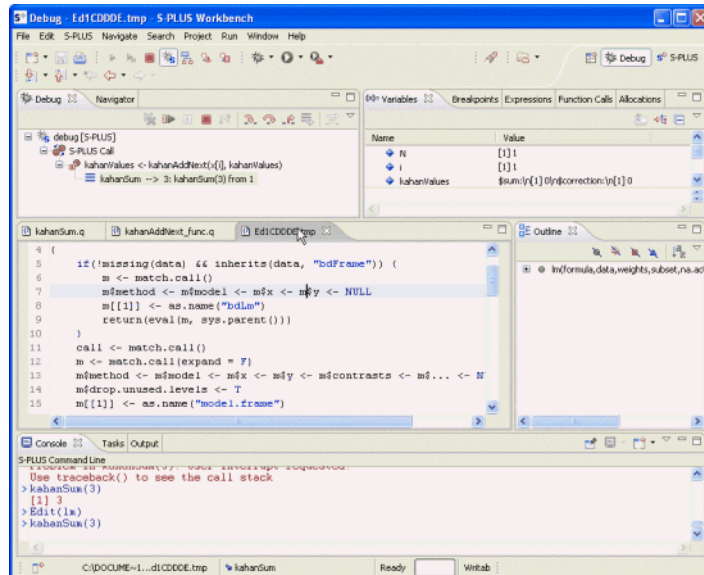




Figure 4.42: *Stepping into a function displayed in a temporary file.*

You can also step over a function () , or you can step out of a function () , using the buttons on the **Debug** view toolbar.


Note

If you are debugging using the **Run** button, rather than running a function via the **Console** view, when you reach the last expression, you can find yourself in internal S-PLUS code. To avoid this situation, type the name of the function and parameters in the **Console** view instead. See the section Debugging Using the Run Button on page 158 for more information.

Examining Resource Usage

You can track resource allocation usage by engaging the S-PLUS Workbench Profiler.

To track resource usage

1. On the S-PLUS Workbench main menu, click the Profiler button () to toggle it on.
2. Click the **Allocations** view tab.
3. Examine the resource usage. Note that it is sorted alphabetically by type. Click the **Type** column head; note that the view is now sorted in reverse.
4. Click the **Amount** column head; note that the view is now sorted by amount, smallest to largest.
5. Click **Amount** again; note that it re-sorts, largest to smallest.
6. Click the drop-down menu, and then click **Reset Allocations**. Note that the table is cleared.

Examining Function Calls

In the next section, examine the functions used in the example. You can examine the functions either in an expandable tree view or in a table view. (Make sure the Profiler is engaged by toggling on the **Profiler** button on the main menu.)

To track functions used

1. Click the **Function Calls** view tab.

2. Examine the tree. Note that after each function, in parentheses, is the amount of time the function call took to run.
3. Scroll down to the `kahanAddNext` function, expand the selection, and examine the time values.

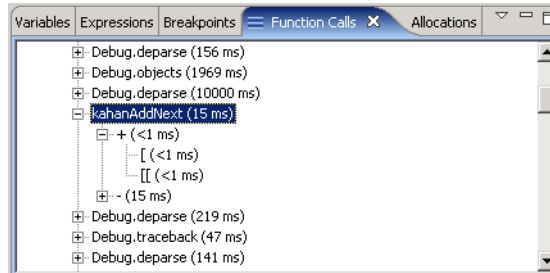


Figure 4.43: Tree view of the **Function Calls** view.

4. Right-click the **Function Calls** view and click **Show Function Tree** to clear it.
5. Examine the resulting table.
6. Scroll down to the `kahanAddNext` function and review the call count (that is, number of times called so far) and duration.

Function	Call Count	Duration (ms)
is.matrix	9	0
is.na	177	1022
is.numeric	305	766
isClipboard...	41	109
isGeneric	59	171
isObject	592	1655
isOldClass	32	5645
isVirtualClass	32	62
javaGuiObje...	9	4781
kahanAddNext	8	122
kahanSum	1	0

Figure 4.44: Table view of the **Function Calls** view.

TROUBLESHOOTING

5

Introduction	154
“Workspace in Use” Error	155
Working with Calls to S-PLUS GUI Functions	156
View is not visible	157
Debugging Using the Run Button	158
Subclipse Add-in error with Workbench	159

INTRODUCTION

This section provides information about using S-PLUS code in the Workbench, and presents workarounds and solutions for special cases you might encounter.

“WORKSPACE IN USE” ERROR

Occasionally, when you start the S-PLUS Workbench, you might see an error that the workspace is already in use when you have not been running a Workbench session.

This problem occurs when you switch computers or versions of UNIX, and you have changes to absolute paths to a given directory.

1. Check your computer to ensure that the path listed in the **Workspace Selection** dialog exists.
2. After you set the appropriate path to an existing workspace, if the problem persists, check for a **.lock** file in the **WORKSPACE/.metadata** directory. Delete this file.
3. If you do not find a **.lock** file, check the running processes to see if old Eclipse processes are running that point to the workspace. End the processes and try re-starting the S-PLUS Workbench.

WORKING WITH CALLS TO S-PLUS GUI FUNCTIONS

While you are working with S-PLUS, you might encounter libraries or sample code that include calls to S-PLUS GUI functions. For example, the **nSurvival** library includes calls to `guiCreate` and `guiRemove` functions in their `.First.lib` and `.Last.lib` objects, respectively.

To solve this problem, in `.First.lib` and `.Last.lib` functions, wrap the code that creates and removes Windows GUI elements in conditional statements. For example:

```
if(interactive() && platform() == "WIN386" &&
  getenv("S_ECLIPSE") == "") {
  ##...Code that creates or removes Windows GUI elements...##
}
```

VIEW IS NOT VISIBLE

If you accidentally close a view, or if the view you want to see is not visible, on the **Windows ► Show View** menu, select the view to display. If the view to display is not on this list, click **Other** to display the **Show View** dialog, and then select the folder, and then the view to display.

DEBUGGING USING THE RUN BUTTON

When you use the **Run** button to invoke a debug session rather than using the **Console** view, and you come to the last expression and click **Step**, **Step In**, or **Step Out**, you can end up in internal S-PLUS functions that are called by your code.

This behavior occurs because any time you click **Run**, the expression that S-PLUS runs is wrapped in a complex S expression designed to capture parse errors, syntax errors, and so on.

To avoid finding yourself in this internal code, run the function by typing it in the **Console** view.

SUBCLIPSE ADD-IN ERROR WITH WORKBENCH

If you are running the Subclipse add-in while you run the S-PLUS Workbench in Microsoft Windows[®], you might see the following error when you switch between projects:

`"Unknown problem executing expression (interrupt?)"`

It is possible to remove this problem by performing an SVN Cleanup operation in the Workbench.

INDEX

Symbols

- .Data
 - working 40
- .Data database 10
- .metadata database 10

A

- add a task
 - in script file 134
- Allocations view 33, 74
- anonymous functions
 - showing in outline 20

B

- background color 18
- background color in Console view
 - changing 17, 21
- Breakpoint
 - types 91
- breakpoints
 - Line 91
- Breakpoints view 33, 74, 89
- Breakpoints view menu
 - Deselect Default Working Set 95
 - Select Default Working Set 95
 - Working Sets 95
- Breakpoints view toolbar 92

C

- changing databases
 - adding a directory 126
 - adding a library 125
 - adding a module 126
- clear History view 55
- code indenting 46
- code problems
 - locating 59
- collapsing breakpoints 93
- color options
 - user-defined 18
- commands
 - persisting history 16
 - scroll through 37
- comparing versions 48
- console fonts 16, 21
- Console view 33, 37, 53, 74
- Console view menu 36, 38
- copy
 - project files between projects 108
- copy history to the Console view 55
- copying code
 - from the Console view 136
- copying from script to console 135
- Copy to Console 25
- create a Workbench project 101
- current working directory 40
- customized menus 23
- customized toolbars 23

D

- databases
 - detaching 61
 - examining search path 60
 - manipulating 56
- debugger 158
- Debug perspective 66
- Debug perspective views 74
- Debug view 74
- Debug view toolbar 77
- Define Folding Region 24
- defining color
 - user terms 18
- deprecated 122
- deselecting default working set
 - Breakpoints view menu 95
- Details pane
 - Expressions view 89
 - Variables view 86
- device
 - default 15
- dialog
 - Filters 59, 63
 - Outline options 20
 - Preferences 12
 - Select Perspective 122
 - Show View 121
 - Sorting 59, 63
 - Task Tags 22
 - Workspace Launcher 101
- directory
 - attaching 61
- down arrow 37

E

- Eclipse 2
- edit code 129
- editing
 - function definitions 130
- editor 45
- empty project
 - creating 102
- existing files

- creating a project for 102
- existing project
 - importing files for 103
- expanding breakpoints 93
- expression evaluating
 - hovering 18, 81
- expressions 146
 - limiting return 85
- Expressions view 74
 - Details pane 89
- Expressions view control menu 84
- Expressions view toolbar 88
- external files
 - opening 49

F

- file associations 12
- files
 - formatting 39
 - opening files not in your project 23
- filtering files 49
- Filters dialog 59, 63
- find
 - function calls 25
- FIXME
 - high-priority tasks 61
- font settings
 - console and output 16, 21
- format code 24
- Format S-Plus Files 39
- Function Calls view 75
- function definition
 - editing 25, 130
- function definitions
 - editing 130
- function help 7
- functions
 - watching 20

G

- Go to File for Breakpoint 93
- Group By 95

H

- help
 - displaying 46
- help menu 26
- high-priority tasks 61
- history
 - persisting 16
- History view 53, 54, 136
- hover 85
- hovering
 - evaluating expression 18, 81

I

- IDE defaults
 - S-PLUS perspective 12
- importing files 103
- Import menu command 108
- indenting
 - code 46
- internal S-PLUS functions
 - in temporary files 158
- interrupt code 45

J

- Java graph 15

L

- library
 - attaching 61
- Line breakpoints 91
- line limits
 - History view 56
- line numbers 129, 130
 - displaying 45
- low-priority tasks 61

M

- medium-priority tasks 61
- menu
 - Console view 36, 38
 - help 26

- Objects view 56
- Problems view 59, 60
- Run 26
- S-PLUS 24
- Tasks view 63
- Window 26

- menus
 - customized 23
- module
 - attaching 61
- multiple projects 106

N

- Navigator view 54, 128
- New Project wizard 23

O

- Object Explorer
 - Workbench 58
- object members
 - changing number displayed in
 - Objects view 58
- objects
 - examining 57
- Objects view 53, 56
- Objects view menu 56
- opening external files 23, 46
- Outline dialog 20
- Outline view 41, 53, 75
- Outline view toolbar 43
- output fonts 16, 21
- Output View 75
- Output view 43, 53

P

- PDF reader
 - specifying 8, 14
- Perspective 3
- perspective 31
 - Debug 66
 - reset 122
 - S-PLUS 52

- Preferences
 - debugging 69
- preferences 12
 - setting 110
- Preferences dialog 12
- Problems view 54, 59, 137
- Problems view menu 59, 60
- profiler 67
- project files
 - copying 108
 - removing 128

R

- refreshing
 - Objects view 57
 - Problems view 59
 - Search Path view 61
 - views 127
- Remove All Breakpoints 92
- removing
 - project files 128
- restoring files 48
- reviewing objects 57
- run 26
- Run Current File 135
- Run menu 26
- run next command 26
- running code 26, 45, 134, 135
 - on startup 14
- running scripts 135
- running S-PLUS code 27, 29

S

- script
 - creating 128
- Script Editor 45
 - in the Debug perspective 75
- searching terms 48
- Search Path View 60
- Search Path view 54, 125
- selecting the default working set
 - Breakpoints view menu 95

- Select Perspective dialog 122
- setting return limits
 - variables and expressions 85
- shared views 33
- show anonymous functions 20
- Show Breakpoints Supported by
 - Selected Target 92
- Show View dialog 121
- simultaneous sessions 2
- Skip All Breakpoints 93
- Sorting dialog 59, 63
- Source S-PLUS Files 39
- specifying a PDF reader 8, 14
- S-PLUS
 - internal functions 158
- S-PLUS (Deprecated) 122
- S-PLUS Debugger
 - toggling 28, 30
- S-PLUS error breakpoint
 - toggling 29, 30
- S-PLUS menu 24
 - Find 25
 - Find References 25
 - Format 24
 - Open S-PLUS Help File 25
 - Run Current File 24
 - Run Selection 24
 - Shift Left 24
 - Shift Right 24
 - Toggle Comment 24
- S-PLUS Packages 8
- S-PLUS perspective 52
- S-PLUS perspective views 53
- S-PLUS Profiler
 - toggling 28, 30
- S-PLUS warning breakpoint
 - toggling 28, 30
- S-PLUS Workbench 2
- starting the Workbench 9
- step into
 - internal S-PLUS code 158
- stop 26
- store console history 16

T

- table pane
 - Objects view 57
- task levels 61
- Tasks view 54
- Tasks view menu 63
- Tasks view toolbar 62
- task tags
 - defining 22
- Task Tags dialog 22
- text variables
 - limiting return 85
- TODO
 - medium-priority tasks 61
- toggle Debug mode 26
- toggle Profile mode 26
- toggle S-PLUS error breakpoint 26
- toggle S-PLUS warning breakpoint 26
- Toggle Working S-PLUS Project 40
- tooggling comments 24
- tooggling S-PLUS Debugger 28, 30
- tooggling S-PLUS error breakpoint 29, 30
- tooggling S-PLUS Profiler 28, 30
- tooggling S-PLUS warning breakpoint 28, 30
- toolbar
 - Breakpoints 92
 - Debug view 77
 - Expressions view 88
 - Outline view 43
 - S-PLUS Workbench 27
 - Tasks view 62
- toolbars
 - customized 23
- tooltip 18, 81
- tree view pane
 - Objects view 58

U

- up arrow 37

V

- Variables view 75, 83
 - Details pane 86
- Variables view control menu 84
- view
 - Allocations 74
 - Breakpoints 74
 - Console 74
 - Debug 74
 - definition 32
 - display issues 122
 - Expressions 74
 - Function Calls 75
 - Outline 75
 - Output 75
 - refreshing 61
 - Search Path 60, 125
 - Variables 75
- views
 - changing display 121
 - customizing 120
 - Debug perspective 74
 - shared 33
 - S-PLUS perspective 53

W

- watching functions 20
- Window menu 26
- Workbench Project 4
- Workbench project
 - creating 101
- Workbench Script Editor 5
- Workbench User Guide 6
- Workbench View 5
- working directory
 - setting current 40
- Working Sets
 - Breakpoints view menu 95
- Workspace 4
- workspace 10, 101
 - changing 101
- Workspace Launcher dialog 101

X

XXX 61